

```

1      title 'CP/M 3 - Console Command Processor - November 1982'
2      ;      version 3.00 Nov 30 1982 - Doug Huskey
3
4
5      ; Copyright (C) 1982
6      ; Digital Research
7      ; P.O. Box 579
8      ; Pacific Grove, CA 93950
9
10     ; Revised: (date/name of person modifying this source)
11
12     ; *****
13     ; ***** The following equates must be set to 100H ***
14     ; ***** + the addresses specified in LOADER.PRN ***
15     ; *****
16     P0100 = equ1 equ    rsxstart ;does this adr match loader's?
17     P01D0 = equ2 equ    fixchain ;does this adr match loader's?
18     P01EB = equ3 equ    fixchain1 ;does this adr match loader's?
19     P01F0 = equ4 equ    fixchain2 ;does this adr match loader's?
20     P0200 = equ5 equ    rsx$chain ;does this adr match loader's?
21     P02CA = equ6 equ    reloc     ;does this adr match loader's?
22     P030F = equ7 equ    calcdst   ;does this adr match loader's?
23     P038D = equ8 equ    scbaddr   ;does this adr match loader's?
24     P038F = equ9 equ    banked    ;does this adr match loader's?
25     P0394 = equ10 equ   rsxend    ;does this adr match loader's?
26     P041A = equ11 equ   ccporg    ;does this adr match loader's?
27     POD80 = equ12 equ   ccpend    ;This should be 0D80h
28     0100 =      rsxstart      equ    0100h
29     01D0 =      fixchain      equ    01D0h
30     01EB =      fixchain1     equ    01EBh
31     01F0 =      fixchain2     equ    01F0h
32     0200 =      rsx$chain     equ    0200h
33     02CA =      reloc         equ    02CAh
34     030F =      calcdst       equ    030Fh
35     038D =      scbaddr       equ    038Dh
36     038F =      banked        equ    038Fh
37     0394 =      rsxend        equ    0394h
38     041A =      ccporg        equ    041Ah
39     ; *****
40     ; NOTE: THE ABOVE EQUATES MUST BE CORRECTED IF NECESSARY
41     ; AND THE JUMP TO START AT THE BEGINNING OF THE LOADER
42     ; MUST BE SET TO THE ORIGIN ADDRESS BELOW:
43
44     041A      org    ccporg      ;LOADER is at 100H to 3??H
45
46     ;      (BE SURE THAT THIS LEAVES ENOUGH ROOM FOR THE LOADER BIT MAP)
47
48
49     ; Conditional Assembly toggles:
50
51     FFFF =      true equ    0ffffh
52     0000 =      false equ   0h
53     FFFF =      newdir equ   true
54     FFFF =      newera equ   true      ;confirm any ambiguous file name
55     FFFF =      dayfile equ   true
56     0000 =      prompts equ   false

```

CP/M-PLUS V3.0

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135

```

57 FFFF = func152 equ true
58 FFFF = multi equ true ;multiple command lines
59 ;also shares code with loader (100-2??h)
60 ;
61 ;*****
62 ;
63 ; GLOBAL EQUATES
64 ;
65 ;*****
66 ;
67 ;
68 ; CP/M BASE PAGE
69 ;
70 0000 = wstart equ 0 ;warm start entry point
71 0004 = defdrv equ 4 ;default user & disk
72 0005 = bdos equ 5 ;CP/M BDOS entry point
73 0006 = osbase equ bdos+1 ;base of CP/M BDOS
74 0050 = cmdrv equ 050h ;command drive
75 005C = dfcb equ 05ch ;1st default fcb
76 005B = dufcb equ dfcb-1 ;1st default fcb user number
77 0051 = pass0 equ 051h ;1st default fcb password addr
78 0053 = len0 equ 053h ;1st default fcb password length
79 006C = dfcb1 equ 06ch ;2nd default fcb
80 006B = dufcb1 equ dfcb1-1 ;2nd default fcb user number
81 0054 = pass1 equ 054h ;2nd default fcb password addr
82 0056 = len1 equ 056h ;2nd default fcb password length
83 0080 = buf equ 80h ;default buffer
84 0100 = tpa equ 100h ;transient program area
85 ; if multi
86 00E7 = comlen equ 100h-19h ;maximum size of multiple command
87 ;RSX buffer with 16 byte header &
88 ;terminating zero
89 ; else
90 comlen equ tpa-buf
91 endif
92 ;
93 ; BDOS FUNCTIONS
94 ;
95 0031 = vers equ 31h ;BDOS vers 3.1
96 0001 = cinf equ 1 ;console input
97 0002 = coutf equ 2 ;console output
98 0006 = craf equ 6 ;raw console input
99 0009 = pbuff equ 9 ;print buffer to console
100 000A = rbuff equ 10 ;read buffer from console
101 000B = cstatf equ 11 ;console status
102 000D = resetf equ 13 ;disk system reset
103 000E = self equ 14 ;select drive
104 000F = openf equ 15 ;open file
105 0010 = closef equ 16 ;close file
106 0011 = searf equ 17 ;search first
107 0012 = searnf equ 18 ;search next
108 0013 = delf equ 19 ;delete file
109 0014 = readf equ 20 ;read file
110 0016 = makef equ 22 ;make file
111 0017 = renf equ 23 ;rename file
112 001A = dmof equ 26 ;set DMA address

```

COPYRIGHT © 1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950
 SER. # _____

```

113 0020 =      userf equ      32          ;set/get user number
114 0021 =      rreadf equ     33          ;read file
115 0030 =      flushf equ     48          ;flush buffers
116 0031 =      scbf equ       49          ;set/get SCB value
117 003B =      loadf equ       59          ;program load
118 0062 =      allocf equ      98          ;reset allocation vector
119 0063 =      trunf equ       99          ;read file
120 0098 =      parsef equ     152          ;parse file
121          ;
122          ;      ASCII characters
123          ;
124 0003 =      ctrlc: equ      'C'-40h
125 000D =      cr: equ        'M'-40h
126 000A =      lf: equ        'J'-40h
127 0009 =      tab: equ       'I'-40h
128 001A =      eof: equ       'Z'-40h
129          ;
130          ;
131          ;      RSX MEMORY MANAGEMENT EQUATES
132          ;
133          ;      RSX header equates
134          ;
135 0006 =      entry          equ      06h          ;RSX contain jump to start
136 000B =      nextadd          equ      0bh          ;address of next RXS in chain
137 000C =      prevadd          equ      0ch          ;address of previous RSX in chain
138 000E =      warnflg          equ      0eh          ;remove on wboot flag
139 0018 =      endchain          equ      18h          ;end of RSX chain flag
140          ;
141          ;      LOADER.RSX equates
142          ;
143 0100 =      module          equ      100h          ;module address
144          ;
145          ;      COM file header equates
146          ;
147 0101 =      comsize          equ      tpa+1h          ;size of the COM file
148 0110 =      rsxoff          equ      tpa+10h          ;offset of the RSX in COM file
149 0112 =      rsxlen          equ      tpa+12h          ;length of the RSX
150          ;
151          ;
152          ;      SYSTEM CONTROL BLOCK OFFSETS
153          ;
154 009C =      pag$off          equ      09ch
155          ;
156 0090 =      olog          equ      pag$off-0ch          ; removeable media open vector
157 0092 =      rlog          equ      pag$off-0ah          ; removeable media login vector
158 0098 =      bdosbase          equ      pag$off-004h          ; real BDOS entry point
159 009C =      hash1          equ      pag$off+000h          ; system variable
160 009D =      hash          equ      pag$off+001h          ; hash code
161 00A1 =      bdos$version          equ      pag$off+005h          ; BDOS version number
162 00A2 =      util$flgs          equ      pag$off+006h          ; utility flags
163 00A6 =      dspl$flgs          equ      pag$off+00ah          ; display flags
164 00AA =      clp$flgs          equ      pag$off+00eh          ; CLP flags
165 00AB =      clp$drv          equ      pag$off+00fh          ; submit file drive
166 00AC =      prog$ret$code          equ      pag$off+010h          ; program return code
167 00AE =      multi$rsx$pg          equ      pag$off+012h          ; multiple command buffer page
168 00AF =      ccpdrv          equ      pag$off+013h          ; ccp default drive

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

169 00B0 =      ccpusr      equ      pag$off+014h      ; ccp default user number
170 00B1 =      ccpconbuf  equ      pag$off+015h      ; ccp console buffer address
171 00B3 =      ccpflag1   equ      pag$off+017h      ; ccp flags byte 1
172 00B4 =      ccpflag2   equ      pag$off+018h      ; ccp flags byte 2
173 00B5 =      ccpflag3   equ      pag$off+019h      ; ccp flags byte 3
174 00B6 =      conwidth   equ      pag$off+01ah      ; console width
175 00B7 =      concolumn  equ      pag$off+01bh      ; console column position
176 00B8 =      conpage    equ      pag$off+01ch      ; console page length (lines)
177 00B9 =      conline    equ      pag$off+01dh      ; current console line number
178 00BA =      conbuffer  equ      pag$off+01eh      ; console input buffer address
179 00BC =      conbuffi   equ      pag$off+020h      ; console input buffer length
180 00BE =      conin$rflg equ      pag$off+022h      ; console input redirection flag
181 00C0 =      conout$rflg equ      pag$off+024h      ; console output redirection flag
182 00C2 =      auxin$rflg equ      pag$off+026h      ; auxillary input redirection flag
183 00C4 =      auxout$rflg equ      pag$off+028h      ; auxillary output redirection flag
184 00C6 =      listout$rflg equ      pag$off+02ah      ; list output redirection flag
185 00C8 =      page$mod   equ      pag$off+02ch      ; page mode flag 0=on, 0ffH=off
186 00C9 =      page$def   equ      pag$off+02dh      ; page mode default
187 00CA =      ctlh$act   equ      pag$off+02eh      ; ctl-h active
188 00CB =      rubout$act equ      pag$off+02fh      ; rubout active (boolean)
189 00CC =      type$ahead equ      pag$off+030h      ; type ahead active
190 00CD =      contran    equ      pag$off+031h      ; console translation subroutine
191 00CF =      con$mode   equ      pag$off+033h      ; console mode (raw/cooked)
192 00D1 =      ten$buffer equ      pag$off+035h      ; 128 byte buffer available
193                                     ; to banked BIOS
194 00D3 =      outdelim   equ      pag$off+037h      ; output delimiter
195 00D4 =      listcp     equ      pag$off+038h      ; list output flag (ctl-p)
196 00D5 =      q$flag     equ      pag$off+039h      ; queue flag for type ahead
197 00D6 =      scbad      equ      pag$off+03ah      ; system control block address
198 00D8 =      dmaad      equ      pag$off+03ch      ; dma address
199 00DA =      seldsk     equ      pag$off+03eh      ; current disk
200 00DB =      info       equ      pag$off+03fh      ; BDOS variable "info"
201 00DD =      resel      equ      pag$off+041h      ; disk reselect flag
202 00DE =      relog      equ      pag$off+042h      ; relog flag
203 00DF =      fx         equ      pag$off+043h      ; function number
204 00E0 =      usrcode    equ      pag$off+044h      ; current user number
205 00E1 =      dcnt       equ      pag$off+045h      ; directory record number
206 00E3 =      searcha    equ      pag$off+047h      ; fcb address for searchn function
207 00E5 =      searchl    equ      pag$off+049h      ; scan length for search functions
208 00E6 =      multcnt    equ      pag$off+04ah      ; multi-sector I/O count
209 00E7 =      errormode  equ      pag$off+04bh      ; BDOS error mode
210 00E8 =      drv0       equ      pag$off+04ch      ; search chain - 1st drive
211 00E9 =      drv1       equ      pag$off+04dh      ; search chain - 2nd drive
212 00EA =      drv2       equ      pag$off+04eh      ; search chain - 3rd drive
213 00EB =      drv3       equ      pag$off+04fh      ; search chain - 4th drive
214 00EC =      tempdrv    equ      pag$off+050h      ; temporary file drive
215 00ED =      patch$flag equ      pag$off+051h      ; patch flags
216 00F4 =      date       equ      pag$off+058h      ; date stamp
217 00F9 =      com$base   equ      pag$off+05dh      ; common memory base address
218 00FB =      error      equ      pag$off+05fh      ; error jump...all BDOS errors
219 00FE =      top$tpa    equ      pag$off+062h      ; top of user TPA (address at 6,7)
220                                     ;
221                                     ;      CCP FLAG 1 BIT MASKS
222                                     ;      (used with getflg, setflg and resetflg routines)
223                                     ;
224 0080 =      chainflg    equ      080h            ; program chain (funct 49)

```

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

225 003F = not$chainflg equ 03fh ; mask to reset chain flags
226 0040 = chainenv equ 040h ; preserve usr/drv for chained prog
227 B320 = comredirect equ 0b320h ; command line redirection active
228 B310 = menu equ 0b310h ; execute ccp.ovl for menu systems
229 B308 = echo equ 0b308h ; echo commands in batch mode
230 B304 = userparse equ 0b304h ; parse user numbers in commands
231 B301 = subfile equ 0b301h ; $$$SUB file found or active
232 0001 = subfilemask equ subfile-0b300h
233 0002 = rsx$only$set equ 02h ; RSX only load (null COM file)
234 00FD = rsx$only$clr equ 0FDh ; reset RSX only flag
235 ;
236 ; CCP FLAG 2 BIT MASKS
237 ; (used with getflg, setflg and resetflg routines)
238 ;
239 B4A0 = ccp10 equ 0b4a0h ; CCP function 10 call (2 bits)
240 B420 = ccpsub equ 0b420h ; CCP present (for SUBMIT, PUT, GET)
241 B480 = ccpbdos equ 0b480h ; CCP present (for B.DOS buffer save)
242 0020 = dskreset equ 20h ; CCP does disk reset on ^C from prompt
243 B440 = submit equ 0b440h ; input redirection active
244 0040 = submitflg equ 40h ; input redirection flag value
245 B418 = order equ 0b418h ; command order
246 ; 0 - COM only
247 ; 1 - COM,SUB
248 ; 2 - SUB,COM
249 ; 3 - reserved
250 B404 = datetime equ 0b404h ; display date & time of load
251 B403 = display equ 0b403h ; display filename & user/drive
252 0002 = filename equ 02h ; display filename loaded
253 0001 = location equ 01h ; display user & drive loaded from
254 ;
255 ; CCP FLAG 3 BIT MASKS
256 ; (used with getflg, setflg and resetflg routines)
257 ;
258 ;
259 0001 = rsxload equ 1h ; load RSX, don't fix chain
260 0002 = coldboot equ 2h ; try to exec profile.sub
261 ;
262 ; CONMODE BIT MASKS
263 ;
264 CF01 = ctlc$stat equ 0cf01h ;conmode CTL-C status
265 ;
266 ;
267 ;
268 ;*****
269 ;
270 ; Console Command Processor - Main Program
271 ;
272 ;*****
273 ;
274 ;
275 ;
276 start:
277 ;
278 041A 312D0F lxi sp,stack
279 041D 210C05 lxi h,ccpret ;push CCPRET on stack, in case of
280 0420 E5 push h ; profile error we will backfile

```

COPYRIGHT ©1982
DIGITAL RESEARCH

P.O. BOX 579

BACHELOR GROVE, CA 93950

SER. #

```

281 0421 116A0D      lxi    d,scbaddr
282 0424 0E31        mvi    c,scbf
283 0426 CD0500      call   bdos
284 0429 228D03      shld   scbaddr      ;save SCB address
285 042C 2EFA        mvi    l,com$base+1
286 042E 7E          mov     a,a      ;high byte of commonbase
287 042F 328F03      sta     banked    ;save in loader
288 0432 2E99        mvi    l,bdosbase+1 ;HL addresses real BDOS page
289 0434 7E          mov     a,a      ;BDOS base in H
290 0435 32A00D      sta     realdos   ;save it for use in XCOM routine
291                  ;
292 0438 3A0700      lda     osbase+1   ;is the LOADER in memory?
293 043B 96          sub     m        ;compare link at 6 with real BDOS
294 043C C25804      jnz     reset$alloc ;skip move if loader already present
295                  ;
296                  ;
297 movldr:
298 043F 019402      lxi     b,rsxend-rsxstart ;length of loader RSX
299 0442 CD0F03      call   calcdst    ;calculate destination and (bias+200h)
300 0445 63          mov     h,e      ;set to zero
301 0446 6B          mov     l,e
302                  ; lxi     h,module-100h ;base of loader RSX (less 100h)
303 0447 CDCA02      call   reloc     ;relocate loader
304 044A 2A0600      lhld   osbase    ;HL = BDOS entry, DE = LOADER base
305 044D 6B          mov     l,e      ;set L=0
306 044E 0E06        mvi    c,6
307 0450 CDAE0B      call   move      ;move the serial number down
308 0453 1E0B        mvi    e,nextadd
309 0455 CDEB01      call   fixchain1
310                  ;
311                  ;
312 reset$alloc:
313 0458 0E62        mvi    c,allocf
314 045A CD0500      call   bdos
315                  ;
316                  ;
317                  ;
318 *****
319                  ;
320                  ; INITIALIZE SYSTEM CONTROL BLOCK
321                  ;
322 *****
323                  ;
324                  ;
325 scbinit:
326                  ;
327                  ; # dir columns, page size & function 9 delimiter
328                  ;
329 045D 06B6        mvi    b,conwidth
330 045F CDEFF0B     call   getbyte
331 0462 3C          inr     a          ;get console width (rel 1)
332 0463 0F          rrc
333 0464 0F          rrc
334 0465 0F          rrc
335 0466 0F          rrc
336 0467 E60F       ani     0fh        ;divide by 16

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

337 0469 11970D    lxi    d,dircols
338 046C 12        stax    d            ;dircols = conwidth/16
339 046D 2EB8      mvi     l,conpage
340 046F 7E        mov     a,m
341 0470 3D        dcr     a            ;subtract 1 for space before prompt
342 0471 13        inx     d
343 0472 12        stax    d            ;pgsize = conpage
344 0473 AF        xra     a
345 0474 13        inx     d
346 0475 12        stax    d            ;line=0
347 0476 3E24      mvi     a,'$'
348 0478 13        inx     d
349 0479 12        stax    d            ;pgmode = nopage (>0)
350 047A 2ED3      mvi     l,outdelim
351 047C 77        mov     m,a        ;set function 9 delimiter
352                ;
353                ;      multisector count, error mode, console mode
354                ;      & BDOS version no.
355                ;
356 047D 2EE6      mvi     l,multcnt
357 047F 3601      mvi     m,l        ;set multisector I/O count = 1
358 0481 23        inx     h            ;,error mode
359 0482 AF        xra     a
360 0483 77        mov     m,a        ;set return error mode = 0
361 0484 2ECF      mvi     l,conmode
362 0486 3601      mvi     m,l        ;set ^C status mode
363 0488 23        inx     h
364 0489 77        mov     m,a        ;zero 2nd conmode byte
365 048A 2EA1      mvi     l,bdos$version
366 048C 3631      mvi     m,vers     ;set BDOS version no.
367                ;
368                ;      disk reset check
369                ;
370 048E 2EB4      mvi     l,ccpflag2
371 0490 7E        mov     a,m
372 0491 E620      ani     dskreset    ;^C at CCP prompt?
373 0493 0E0D      mvi     c,resetf
374 0495 E5        push    h
375 0496 C40500    cnz     bdos        ;perform disk reset if so
376 0499 E1        pop     h
377                ;
378                ;      remove temporary RSXs (those with remove flag on)
379                ;
380                ;
381 049A 2EB3      rsxck: mvi     l,ccpflag1    ;check CCP flag for RSX only load
382 049C 7E        mov     a,m
383 049D E602      ani     rsx$only$set ;bit = 1 if only RSX has been loaded
384 049F E5        push    h
385 04A0 C0002     cz      rsx$chain    ;don't fix-up RSX chain if so
386 04A3 E1        pop     h
387 04A4 7E        mov     a,m
388 04A5 E6FD      ani     rsx$only$clr ;clear RSX only loader flag
389 04A7 77        mov     m,a        ;replace it
390                ;
391                ;      chaining environment
392                ;

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

393 04A8 E640      ani    chain$env      ;non-zero if we preserve programs
394 04AA E5        push   h              ;user & drive for next transient
395                ;
396                ;      user number
397                ;
398 04AB 2EB0      mvi     l,ccpusr      ; HL = .CCP USER (saved in SCB)
399 04AD 01700D    lxi     b,usenum      ; BC = .CCP'S DEFAULT USER
400 04B0 54        mov     d,h
401 04B1 1EE0      mvi     e,usrcode     ; DE = .BDOS USER CODE
402 04B3 1A        ldax   d
403 04B4 02        stax   b              ; usenum = bdos user number
404 04B5 7E        mov     a,m           ; ccp user
405 04B6 C2BA04    jnz     scb1          ; jump if chaining env preserved
406 04B9 02        stax   b              ; usenum = ccp default user
407 04BA 12        scb1: stax   d         ; bdos user = ccp default user
408                ;
409                ;      transient program's current disk
410                ;
411 04BB 03        inx     b              ;.CHAINDSK
412 04BC 1EDA      mvi     e,seldsk      ;.BDOS CURRENT DISK
413 04BE 1A        ldax   d
414 04BF C2C404    jnz     scb2          ; jump if chaining env preserved
415 04C2 3EFF      mvi     a,0ffh
416                ;      ; make an invalid disk
417 04C4 02        scb2: stax   b         ; chaindisk = bdos disk (or invalid)
418                ;
419                ;      current disk
420                ;
421 04C5 2B        dcx     h              ;.CCP's DISK (saved in SCB)
422 04C6 03        inx     b              ;.CCP's CURRENT DISK
423 04C7 7E        mov     a,m
424 04C8 02        stax   b
425 04C9 12        stax   d              ; BDOS current disk
426                ;
427                ;      $$$SUB drive
428                ;
429 04CA 2EEC      mvi     l,tempdrv
430 04CC 03        inx     b              ;.SUBFCB
431 04CD 7E        mov     a,m
432 04CE 02        stax   b              ; $$$SUB drive = temporary drive
433                ;
434                ;      check for program chain
435                ;
436 04CF E1        pop     h              ;HL =.ccpflag1
437 04D0 7E        mov     a,m
438 04D1 E680      ani     chainflg      ;is it a chain function (47)
439 04D3 CAE704    jz      ckboot        ;jump if not
440 04D6 218000    lxi     h,buf
441 04D9 11F50D    chain: lxi     d,cbufl
442 04DC 0E7F      mvi     c,tpa-buf-1
443 04DE 79        mov     a,c
444 04DF 12        stax   d
445 04E0 13        inx     d
446 04E1 CDAE0B    call    move          ;hl = source, de = dest, c = count
447 04E4 C3BB05    jmp     ccpparse
448                ;

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____


```

449      ;      execute profile.sub ?
450      ;
451 04E7 2EB5      ckboot:      mvi      1,ccpflag3
452 04E9 7E      mov      a,m
453 04EA E602      ani      coldboot      ;is this a cold start
454 04EC C20605      jnz      ccpcr      ;jump if not
455 04EF 7E      mov      a,m
456 04F0 F602      ori      coldboot      ;set flag for next time
457 04F2 77      mov      m,a
458 04F3 32670D      sta      errflg      ;set to ignore errors
459 04F6 21FC04      lxi      h,profile
460 04F9 C3D904      jmp      chain      ;attempt to exec profile.sub
461      profile:
462 04FC 50524F4649      db      'PROFILE.S',0
463      ;
464      ;
465      ;
466      ;*****
467      ;
468      ;      BUILT-IN COMMANDS (and errors) RETURN HERE
469      ;
470      ;*****
471      ;
472      ;
473      ccpcr:
474      ;      enter here on each command or error condition
475 0506 CDE40B      call      setccpflg
476 0509 CD090C      call      crlf
477      ccpret:
478 050C 212B0F      lxi      h,stack-2      ;reset stack in case of error
479 050F F9      sphl      ;preserve CCPRET on stack
480 0510 AF      xra      a
481 0511 32990D      sta      line
482 0514 210C05      lxi      h,ccpret      ;return for next builtin
483 0517 E5      push      h
484 051B CDE40B      call      setccpflg
485 051B 2B      dcx      h      ;,CCPFLAG1
486 051C 7E      mov      a,m
487 051D E601      ani      subfilemask      ;check for $$$SUB submit
488 051F CA6405      jz      prompt
489      ;
490      ;
491      ;
492      ;*****
493      ;
494      ;      $$$SUB file processing
495      ;
496      ;*****
497      ;
498      ;
499 0522 11F50D      lxi      d,cbufl      ;set DMA to command buffer
500 0525 CD7B09      call      setbuf
501 0528 0E0F      mvi      c,openf
502 052A CDF009      call      sudos      ;open it if flag on
503 052D 0E0B      mvi      c,cstatf      ;check for break if successful open
504 052F CCF009      cz      sudos      ;^C typed?

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. ==

```

505 0532 C24F05      jnz      subclose      ;delete $$$S. if break or open failed
506 0535 21960D      lxi      h,subrr2
507 0538 77          mov      m,a          ;zero high random record #
508 0539 2B          dcx      h
509 053A 77          mov      m,a          ;zero middle random record #
510 053B 2B          dcx      h
511 053C E5          push     h
512 053D 3A820D      lda      subrc
513 0540 3D          dcr      a
514 0541 77          mov      m,a          ;set to read last record of file
515 0542 0E21        mvi      c,rreadf
516 0544 F4F009      cp       sudos
517 0547 E1          pop      h
518 0548 35          dcr      m          ;record count (truncate last record)
519 0549 0E13        mvi      c,delf
520 054B FCF009      cm       sudos
521 054E B7          ora      a          ;error on read
522                  ;
523                  ;
524                  subclose:
525 054F F5          push     psw
526 0550 0E63        mvi      c,trunf      ;truncate file (& close it)
527 0552 CDF009      call     sudos
528 0555 F1          pop      psw          ;any errors ?
529 0556 CA8B05      jz       ccpparse      ;parse command if not
530                  ;
531                  ;
532                  subkill:
533 0559 0101B3      lxi      b,subfile
534 055C CDF10B      call     resetflg      ;turn off submit flag
535 055F 0E13        mvi      c,delf
536 0561 CDF009      call     sudos          ;kill submit
537                  ;
538                  ;
539                  ;
540                  ;*****
541                  ;
542                  ;   GET NEXT COMMAND
543                  ;
544                  ;*****
545                  ;
546                  ;
547                  ;
548                  ;   prompt user
549                  ;
550                  prompt:
551 0564 3A700D      lda      usernum
552 0567 B7          ora      a
553 0568 C4130C      cnz      pdb          ;print user # if non-zero
554 056B CD6D06      call     dirdrv1
555 056E 3E3E        mvi      a,'>'
556 0570 CD1609      call     putc
557                  ;
558                  ;   if multi
559                  ;move ccpconbuf addr to conbuffer addr
560 0573 11BAB1      lxi      d,ccpconbuf*256+conbuffer

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

561 0576 CDA70B      call    wordmov      ;process multiple command, unless in submit
562 0579 B7          ora      a              ;non-zero => multiple commands active
563 057A F5          push    psw            ;save A=high byte of ccpconbuf
564 057B 0180B4       lxi      b,ccpbdos
565 057E C4F10B       cnz      resetflg      ;turn off BDOS flag if multiple commands
566                  endif
567 0581 CD4E09       call    rcIn          ;get command line from console
568 0584 CDEE0B       call    resetccpflg     ;turn off BDOS, SUBMIT & GET ccp flags
569                  if multi
570 0587 F1           pop     psw            ;D=high byte of ccpconbuf
571 0588 C4A80A       cnz      multisave     ;save multiple command buffer
572                  endif
573                  ;
574                  ;
575                  ;
576                  ;*****
577                  ;
578                  ;   PARSE COMMAND
579                  ;
580                  ;*****
581                  ;
582                  ;
583      ccpparse:
584                  ;
585                  ;       reset default page mode
586                  ;       (in case submit terminated)
587                  ;
588 058B CDDA0B       call    subtest      ;non-zero if submit is active
589 058E C29605       jnz      get$pg$mode ;skip, if so
590      set$pg$mode:
591 0591 2EC9         mvi      l,page$def
592 0593 7E          mov     a,m           ;pick up default
593 0594 2B          dcx     h
594 0595 77          mov     m,a          ;place in mode
595      get$pg$mode:
596 0596 2ECB         mvi      l,page$mode
597 0598 7E          mov     a,m
598 0599 329A0D       sta     pgmode
599                  ;
600                  ;check for multiple commands
601                  ;convert to upper case
602                  ;reset ccp flag, in case entered from a CHAIN (or profile)
603                  ;
604 059C CDF609       call    uc            ;convert to upper case, ck if multiple command
605 059F C8          rz                    ;get another line if null or comment
606                  ;
607                  ;transient or built-in command?
608                  ;
609 05A0 11AC0D       lxi      d,u$fcB      ;include user number byte in front of FCB
610 05A3 CD390B       call    gcnd          ;parse command name
611 05A6 3AB60D       lda      fcb+9        ;file type specified?
612 05A9 FE20         cpi      ' '
613 05AB C20406       jnz      ccpdisk2     ;execute from disk, if so
614 05AE 21AC0D       lxi      h,u$fcB      ;user or drive specified?
615 05B1 7E          mov     a,m           ;user number
616 05B2 23          inx     h

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

617 05B3 B6          ora    a          ;drive
618 05B4 23          inx    h
619 05B5 7E          mov    a,m          ;get 1st character of filename
620 05B6 C20A06       jnz    ccpdisk3    ;jump if so
621                  ;
622                  ;BUILT-IN HANDLER
623                  ;
624                  ccpbuiltin:
625 05B9 213706       lxi    h,ctbl      ;search table of internal commands
626 05BC 11AE0D       lxi    d,fcbl+1
627 05BF 3AB00D       lda    fcb+3
628 05C2 FE21        cpi    ' '+1      ;is it shorter than 3 characters?
629 05C4 D4BFOC       cnc    tbls      ;is it a built-in?
630 05C7 C2E805       jnz    ccpdisk0    ;load from disk if not
631 05CA 3AA10D       lda    option     ;[ in command line?
632 05CD B7           ora    a          ;options specified?
633 05CE 78           mov    a,b        ;built-in index from tbls
634 05CF 2A6C0D       lhld   parsep
635 05D2 229D0D       shld   errsav     ;save beginning of command tail
636 05D5 215A06       lxi    h,ptbl      ;jump to processor if options not
637 05D8 CA5808       jz     tblj       ;specified
638 05DB FE04        cpi    4
639 05DD DA8607       jc     trycom
640 05E0 21B10D       lxi    h,fcbl+4
641 05E3 C2EB05       jnz    ccpdisk0    ;if DIRS then look for DIR.COM
642 05E6 3620        mvi    m,' '
643                  ;
644                  ;LOAD TRANSIENT (file type unspecified)
645                  ;
646                  ccpdisk0:
647 05E8 0118B4       lxi    b,order
648 05EB CDD00B       call   getflg     ;0=COM 8=COM,SUB 16=SUB,COM
649 05EE CA0406       jz     ccpdisk2    ;search for COM file only
650 05F1 0608        mvi    b,8        ;=> 2nd choice is SUB
651 05F3 90          sub    b          ;now a=0 (COM first) or 8 (SUB first)
652 05F4 CAF905       jz     ccpdisk1    ;search for COM first then SUB
653 05F7 0600        mvi    b,0        ;search for SUB first then COM
654
655                  ccpdisk1:
656 05F9 C5           push   b          ;save 2nd type to try
657 05FA CD7A0B       call   settype     ; A = offset of type in type table
658 05FD CDE407       call   exec       ;try to execute, return if unsuccessful
659 0600 F1          pop    psw        ;try 2nd type
660 0601 CD7A0B       call   settype
661                  ;
662                  ;LOAD TRANSIENT (file type specified)
663                  ;
664                  ccpdisk2:
665 0604 CDE407       cal_   exec
666 0607 C33A0C       jmp    perror     ;error if can't find it
667                  ;
668                  ;DRIVE SPECIFIED (check for change drives/users command)
669                  ;
670                  ccpdisk3:
671 060A FE20        cpi    ' '          ;check for filename
672 060C C2EB05       jnz    ccpdisk0    ;execute from disk if specified

```

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

673 060F CD360C      call    eoc          ;error if not end of command
674 0612 3AAC0D      lda      ufcB       ;user specified?
675 0615 D601        sui      1
676 0617 DA2506      jc       ccpdrive
677
678                ccpuser:
679 061A 32700D      sta      usernum     ;CCP's user number
680 061D 06B0        mvi      b,ccpusr
681 061F CDF90B      call    setbyte     ;save it in SCB
682 0622 CD8609      call    setuser     ;set current user
683
684                ccpdrive:
685 0625 3AAD0D      lda      fcb         ;drive specified?
686 062B 3D          dcr      a
687 0629 F8          rm              ;return if not
688 062A F5          push     psw
689 062B CD8009      call    select
690 062E F1          pop      psw
691 062F 32720D      sta      disk        ;CCP's drive
692 0632 06AF        mvi      b,ccpdrv
693 0634 C3F90B      jmp      setbyte     ;save it in SCB
694
695                ;;
696                ;
697                ;*****
698                ;
699                ;      BUILT-IN COMMANDS
700                ;
701                ;*****
702                ;
703                ;
704                ;      Table of internal ccp commands
705                ;
706                ;
707 0637 44495220      ctbl:  db      'DIR '
708 063B 5459504520    db      'TYPE '
709 0640 4552415345    db      'ERASE '
710 0646 52454E414D    db      'RENAME '
711 064D 4449525359    db      'DIRSYS '
712 0654 5553455220    db      'USER '
713 0659 00            db      0
714                ;
715 065A 7506          ptbl:  dw      dir
716 065C F406          dw      type
717 065E 2207          dw      era
718 0660 5107          dw      ren
719 0662 7D06          dw      dirs
720 0664 1507          dw      user
721                ;;
722                ;;-----
723                ;;
724                ;;      DIR Command
725                ;;
726                ;;      DIR          list directory of current default user/drive
727                ;;      DIR <X>:    list directory of user/drive <X>
728                ;;      DIR <AFN>    list all files on the current default user/drive

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

729      ;;          with names that match <AFN>
730      ;;          DIR <X>:<AFN> list all files on user/drive <X> with names that
731      ;;          match <AFN>
732      ;;
733      ;;-----
734      ;;
735      ;
736      if newdir
737      dirdrv:
738 0666 3A5C00      lda      dfcb      ;get disk number
739      endif
740
741      dirdrv0:
742 0669 3D          dcr      a
743 066A F27006      jp      dirdrv2
744
745      dirdrv1:
746 066D 3A720D      lda      disk      ;get current disk
747      dirdrv2:
748 0670 C641          adi      'A'
749 0672 C3A60C      jmp      pfc      ;print it (save BC,DE)
750      ;
751      ;
752      if newdir
753      dir:
754 0675 0E00          mvi      c,0      ;flag for DIR (normal)
755 0677 11520D        lxi      d,sysfiles
756 067A C38206        jmp      dirs1
757      ;
758      ;
759      dirs:
760 067D 0E80          mvi      c,080h   ;flag for DIRS (system)
761 067F 114E0D        lxi      d,dirfiles
762
763 0682 D5            dirs1: push    d
764 0683 CD9906        call    direct
765 0686 D1            pop     d      ;de = .system files message
766 0687 CAB807        jz      nofile ;jump if no files found
767 068A 7D            mov     a,l    ;A = number of columns
768 068B B8            cmp     b      ;did we print any files?
769 068C D4090C        cnc      crlf  ;print crlf if so
770 068F 214D0D        lxi      h,anyfiles
771 0692 35            dcr      a
772 0693 34            inr      a
773 0694 C8            rz            ;return if no files
774                                ;except those requested
775 0695 35            dcr      a      ;set to zero
776 0696 C3050C        jmp     pasgnl  ;tell the operator other files exist
777      ;
778      ;
779      direct:
780 0699 C5            push    b      ;save DIR/DIRS flag
781 069A CD7809        call    sbuf80 ;set DMA = 80h
782 069D CDD80A        call    gfn    ;parse file name
783 06A0 115D00        lxi      d,dfcb+1
784 06A3 1A            ldax    d

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

785 06A4 FE20      cpi      ' '
786 06A6 060B      mvi      b,11
787 06AB CCB50C     cz       setmatch      ;use '????????' if none
788 06AB CD360C     call     eoc           ;make sure there's nothing else
789 06AE CDC309     call     srchf        ;search for first directory entry
790 06B1 C1         pop      b
791 06B2 C8         rz           ;if no files found
792               dir0:
793 06B3 3A970D     lda       dircols     ;number of columns for dir
794 06B6 6F         mov      l,a
795 06B7 47         mov      b,a
796 06B8 04         inr      b           ;set # names to print per line (+1)
797               dir1:
798 06B9 E5         push     h           ;L=#cols, B=curent col, C=dir/dirs
799 06BA 210A00     lxi      h,10        ;get byte with SYS bit
800 06BD 19         dad      d
801 06BE 7E         mov      a,m
802 06BF E1         pop      h
803 06C0 E680     ani      80h          ;look at SYS bit
804 06C2 B9         cmp      c          ;DIR/DIRS flag in C
805 06C3 CACE06     jz       dir2        ;display, if modes agree
806 06C6 3E01     mvi      a,1         ;set anyfiles true
807 06C8 324D0D     sta      anyfiles
808 06CB C3E506     jmp      dir3        ;don't print anything
809               ;
810               ; display the filename
811               ;
812               dir2:
813 06CE 05         dcr      b
814 06CF CC0B0C     cz       dirln       ;sets no. of columns, puts crlf
815 06D2 78         mov      a,b        ;number left to print on line
816 06D3 BD         cmp      l         ;is current col = number of cols
817 06D4 CC6606     cz       dirdrv     ;display the drive, if so
818 06D7 3E3A     mvi      a,';'
819 06D9 CDA60C     call     pfc         ;print colon
820 06DC CDA40C     call     space      ;print file name
821 06DF CD8D0C     call     pfn         ;pad with space
822 06E2 CDA40C     call     space
823               dir3:
824 06E5 C5         push     b           ;save current col(B), DIR/DIRS(C)
825 06E6 E5         push     h           ;save number of columns(L)
826 06E7 CD6609     call     break      ;drop out if keyboard struck
827 06EA CDC809     call     srchn      ;search for another match
828 06ED E1         pop      h
829 06EE C1         pop      b
830 06EF C2B906     jnz     dir1
831               direx:
832 06F2 3C         inr      a           ;clear zero flag
833 06F3 C9         ret
834
835               else
836
837               dirs: ; display system files only
838               mvi      a,0d2h        ; JNC instruction
839               sta      dir1         ; skip on non-system files
840               ;

```

CP/M - PLUS V3.0

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135

```

841      dir:  ; display non-system files only
842          lxi    h,ccpcr
843          push   h          ; push return address
844          call   gfn        ;parse file name
845          inx     d
846          ldax   d
847          cpi     ' '
848          mvi     b,11
849          cz      setmatch   ;use '????????' if none
850          call   eoc        ;make sure there's nothing else
851          call   findone    ;search for first directory entry
852          jz      dir4
853          mvi     b,5        ;set # names to print per line
854      dir1: lxi     h,10      ;get byte with SYS bit
855          dad     d
856          mov     a,m
857          ral     ;look at SYS bit
858      dir11: jc      dir3     ;don't print it if SYS bit set
859          mov     a,b
860          push   b
861      dir2: lxi     h,9      ;get byte with R/O bit
862          dad     d
863          mov     a,m
864          ral     ;look at R/O bit
865          mvi     a,' '     ;print space if not R/O
866          jnc     dir21     ;jump if not R/O
867          mvi     a,'*'     ;print star if R/O
868      dir21: call   pfc      ;print character
869          call   pfn        ;print file name
870          mvi     a,13      ;figure out how much padding is needed
871          sub     c
872      dir25: push   psw
873          call   space      ;pad it out with spaces
874          pop     psw
875          dcr     a
876          jnz     dir25     ;loop if more required
877          pop     b
878          dcr     b         ;decrement # names left on line
879          jnz     dir3
880          call   crlf      ;go to new line
881          mvi     b,5      ;set # names to print on new line
882      dir3: push   b
883          call   break      ;drop out if keyboard struck
884          call   srchn      ;search for another match
885          pop     b
886          jnz     dir1
887
888      dir4: mvi     a,0dah   ;JC instruction
889          sta     dir11     ;restore normal dir mode (skip system files)
890          jmp     ccpcr
891
892      endif
893
894      ;;
895      ;;-----
896      ;;

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #


```

897      ;;      TYPE command
898      ;;
899      ;;      TYPE <UFN>      Print the contents of text file <UFN> on
900      ;;                      the console.
901      ;;
902      ;;-----
903      ;;
904      06F4 210605      type: lxi--   h,ccpcr
905      06F7 E5          push    h          ;push return address
906      06F8 CDD80A      call    getfn      ;get and parse filename
907      06FB 3E7F        mvi     a,127      ;initialize buffer pointer
908      06FD 329F0D      sta     bufp
909      0700 0E0F        mvi     c,openf
910      0702 CD6C07      call    sbdosf      ;open file if a filename was typed
911      0705 CD6609      type1: call    break ;exit if keyboard struck
912      0708 CDC40B      call    getb       ;read byte from file
913      070B C0          rnz          ;exit if physical eof or read error
914      070C FE1A        cpi     eof       ;check for eof character
915      070E C8          rz          ;exit if so
916      070F CD1609      call    putc       ;print character on console
917      0712 C30507      jmp     type1      ;loop
918      ;
919      ;;-----
920      ;;
921      ;;      USER command
922      ;;
923      ;;      USER <NN>      Set the user number
924      ;;
925      ;;-----
926      ;;
927      user:
928      0715 11F30C      lxi     d,unmsg      ;Enter User #:
929      0718 CDAB07      call    getprm
930      071B CD520C      call    gdn         ;convert to binary
931      071E C8          rz          ;return if nothing typed
932      071F C31A06      jmp     ccpuser     ;set user number
933      ;
934      ;;-----
935      ;;
936      ;;      ERA command
937      ;;
938      ;;      ERA <AFN>      Erase all file on the current user/drive
939      ;;                      which match <AFN>.
940      ;;      ERA <X>:<AFN>   Erase all files on user/drive <X> which
941      ;;                      match <AFN>.
942      ;;
943      ;;-----
944      ;;
945      0722 CDD80A      era:  call    getfn      ;get and parse filename
946      0725 CA4C07      jz     era1          ;is it ambiguous?
947      0728 CD9E07      call    ckafn
948      072B C24C07      jnz     era1
949      072E 11140D      lxi     d,eramsg
950      0731 CD4909      call    pmsg
951      0734 2A9B0D      lhld    errorp
952      0737 0E20        mvi     c,' '      ;stop at exclamation mark or 0

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

953 0739 CD2A0C      call    pstrg      ;echo command
954 073C 111B0D      lxi      d,confirm
955 073F CD4109      call    getc
956 0742 CD090C      call    crlf
957 0745 7D          mov     a,l        ;character in L after CRLF routine
958 0746 E65F        ani      5fh       ;convert to U/C
959 0748 FE59        cpi      'Y'       ;Y (yes) typed?
960 074A C0          rnz         ;return, if not
961 074B B7          ora      a         ;reset zero flag
962 074C 0E13      eral: mvi      c,delf
963 074E C36C07      jmp      sbdosf
964
965 ;;-----
966 ;;
967 ;;
968 ;;      REN command
969 ;;
970 ;;-----
971 ;;
972 0751 CDD80A      ren:  call    gfn      ;zero flag set if nothing entered
973 0754 F5          push    psw
974 0755 211000      lxi      h,16
975 0758 19          dad     d
976 0759 EB          xchg
977 075A D5          push    d          ;DE = .dfcb+16
978 075B E5          push    h          ;HL = .dfcb
979 075C 0E10        mvi      c,16
980 075E CDAE0B      call    move      ;DE = dest, HL = source
981 0761 CDD80A      call    gfn
982 0764 E1          pop     h          ;HL=.dfcb
983 0765 D1          pop     d          ;DE=.dfcb+16
984 0766 CD9207      call    drvok
985 0769 0E17        mvi      c,renf     ;make rename call
986 076B F1          pop     psw       ;zero flag set if nothing entered
987
988 ;;-----
989 ;;
990 ;;      BUILT-IN COMMAND BDOS CALL & ERROR HANDLERS
991 ;;
992 ;;-----
993 ;
994 sbdosf:
995 076C F5          push    psw
996 076D C4360C      cnz      eoc        ;make sure there's nothing else
997 0770 F1          pop     psw
998 0771 115C00      lxi      d,dfcb
999 0774 06FF        mvi      b,0ffh
1000 0776 2601        mvi      h,1        ;execute disk command if we don't call
1001 0778 C49109      cnz      bdosf     ;call if something was entered
1002 077B C0          rnz         ;return if successful
1003
1004 ferror:
1005 077C 25          dcr     h          ;was it an extended error?
1006 077D FAB807      jm      nofile
1007 0780 2A9D0D      lhld    errsav
1008 0783 226C0D      shld    parsep

```

COPYRIGHT © 1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

1009 0786 CDE407 trycom: call exec
1010 0789 CD8DOC      call pfn
1011 078C 110A0D      lxi d,required
1012 078F C3BB07      jmp builtin$err
1013
1014
1015
1016
1017 ; check for drive conflict
1018 ; HL = FCB
1019 ; DE = FCB+16
1020
1021 0792 1A drvok: ldax d ;get byte from 2nd fcb
1022 0793 BE      cmp m ;ok if they match
1023 0794 CB      rz
1024 0795 B7      ora a ;ok if 2nd is 0
1025 0796 CB      rz
1026 0797 34      inr m ;error if the 1st one's not 0
1027 0798 35      dcr m
1028 0799 C23A0C   jnz perror
1029 079C 77      mov m,d ;copy from 2nd to 1st
1030 079D C9      ret
1031
1032
1033 ; check for ambiguous reference in file name/type
1034
1035 ; entry: b = length of string to check (ckafn0)
1036 ; de = fcb area to check (ckafn0) - 1
1037 ; exit: z = set if any ? in file reference (ambiguous)
1038 ; z = clear if unambiguous file reference
1039
1040 ckafn:
1041 079E 060B      mvi b,11 ;check entire name and type
1042 07A0 13 ckafn0: inx d
1043 07A1 1A      ldax d
1044 07A2 FE3F      cpi '?' ;is it an ambiguous file name
1045
1046 07A4 CB      rz ;return true if any afn
1047
1048      else
1049      rnz ;return true only if *.*
1050      endif
1051 07A5 05      dcr b
1052 07A6 C2A007   jnz ckafn0
1053 07A9 05      dcr b ;clear zero flag to return false
1054      endif
1055 07AA C9      ret ;remove above DCR to return true
1056
1057
1058
1059 ; get parameter (generally used to get a missing one)
1060
1061 getprm:
1062 07AB CD410A   call skps ;see if already there
1063 07AE C0      rnz ;return if so
1064
getp0:

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

1065      if prompts
1066      push    d
1067      lxi     d,enter
1068      call    pmsg
1069      pop     d
1070      endif
1071 07AF CD4909      call    pmsg      ;print prompt
1072 07B2 CD4E09      call    rcln      ;get response
1073 07B5 C3F609      jmp     uc      ;convert to upper case
1074      ;
1075      ;;
1076      ;;-----
1077      if      not newdir
1078      ;;
1079      ;;      search for first file, print "No File" if none
1080      ;;
1081      indone:
1082      call    srchf
1083      rnz          ;found
1084      endif
1085      ;;-----
1086
1087      nofile:
1088 07B8 11020D      lxi     d,nomsg      ;tell user no file found
1089      builtin$err:
1090 07BB CD050C      call    pmsgnl
1091 07BE C30C05      jmp     ccpret
1092
1093      ;
1094      ;
1095      ;*****
1096      ;
1097      ;      EXECUTE DISK RESIDENT COMMAND
1098      ;
1099      ;*****
1100      ;
1101      ;
1102 07C1 005355424Dx fcb: db      0,'SUBMIT COM' ;processor fcb
1103      ;
1104      ;
1105      ;      execute submit file (or any other processor)
1106      ;
1107      xsub:          ;DE = .fcb
1108 07CD 1A          ldax    d
1109 07CE 06AB          mvi     b,clp$drv
1110 07D0 CDF90B      call    setbyte      ;save submit file drive
1111 07D3 21C107      lxi     h,x fcb
1112 07D6 0E0C          mvi     c,l2
1113 07D8 CDAE0B      call    move      ;copy processor into fcb
1114 07DB 21F50D      lxi     h,cbufl      ;set parser pointer back to beginning
1115 07DE 3620          mvi     m,' '
1116 07E0 23          inx     h      ;move past blank
1117 07E1 226C0D      shld    parsep
1118      ;      execute SUBMIT.COM
1119      ;
1120      ;

```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

1121      ;      execute disk resident command (return if not found or error)
1122      ;
1123      exec1:
1124      ;try to open and execute fcb
1125      07E4 11B60D      lxi      d,fcbl+9
1126      07E7 216308      lxi      h,typtbl
1127      07EA CDBF0C      call     tbls      ;search for type in type table
1128      07ED C0          rnz          ;return if no match
1129      07EE 11AC0D      lxi      d,ufcb
1130      07F1 1A          ldax      d      ;check to see if user specified
1131      07F2 B7          ora      a
1132      07F3 C0          rnz          ;return if so
1133      07F4 13          inx      d
1134      07F5 1A          ldax      d      ;check if drive specified
1135      07F6 4F          mov      c,a
1136      07F7 C5          push     b      ;save type (B) and drive (C)
1137      07F8 0E00        mvi      c,0      ;try only 1 open if drive specified
1138      07FA B7          ora      a
1139      07FB C22108      jnz      exec1      ;try to open as specified
1140      07FE 0104E7      lxi      b,(drv0-1)*256+4;try upto four opens from drv chain
1141      0801 3A720D      lda      disk
1142      0804 3C          inr      a
1143      0805 67          mov      h,a      ;save default disk in H
1144      0806 2E01        mvi      l,1      ;allow only 1 match to default disk
1145      0808 04          exec0: inr      b      ;next drive to try in SCB drv chain
1146      0809 0D          dcr      c      ;any more tries?
1147      080A 79          mov      a,c
1148      080B E5          push     h
1149      080C F4FF0B      cp      getbyte
1150      080F E1          pop      h
1151      0810 B7          ora      a
1152      0811 FA7608      jm      exec3
1153      0814 CA1B08      jz      exec01      ;jump if drive is 0 (default drive)
1154      0817 BC          cmp      h      ;is it the default drive
1155      0818 C22008      jnz      exec02      ;jump if not
1156      081B 7C          exec01: mov      a,h      ;set drive explicitly
1157      081C 2D          dcr      l      ;is it the 2nd reference
1158      081D FA080B      jm      exec0      ;skip, if so
1159      0820 12          exec02: stax      d      ;put drive in FCB
1160      0821 C5          exec1: push     b      ;save drive offset(B) & count(C)
1161      0822 E5          push     h
1162      0823 CD8B09      call     opencom      ;on default drive & user
1163      0826 E1          pop      h
1164      0827 C1          pop      b
1165      0828 CA0808      jz      exec0      ;try next if open unsuccessful
1166      ;
1167      ;      successful open, now jump to processor
1168      ;
1169      exec2:
1170      if      dayfile
1171      082B 0103B4      lxi      b,display
1172      082E CDD00B      call     getflg
1173      0831 CA5408      jz      exec21
1174      0834 1A          ldax      d
1175      0835 CD6906      call     dirdrv0
1176      0838 3E3A        mvi      a,'!'

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950
 SER. # _____

```

1177 083A CDA60C      call    pfc
1178 083D D5          push    d
1179 083E CD8D0C      call    pfn
1180 0841 D1          pop     d
1181 0842 D5          push    d
1182 0843 210800      lxi     h,8
1183 0846 19          dad     d
1184 0847 7E          mov     a,m
1185 0848 E680        ani     80h
1186 084A 11420D      lxi     d,userzero
1187 084D C44909      cnz     pasg
1188 0850 CD090C      call    crlf
1189 0853 D1          pop     d
1190                endif
1191 0854 F1          exec21: pop     psw          ;recover saved command type
1192 0855 217008      lxi     h,xptbl
1193                ;
1194                ; table jump
1195                ;
1196                ; entry: hl = address of table of addresses
1197                ;      a = entry # (0 thru n-1)
1198                ;
1199 0858 87          tblj: add     a          ;adjust for two byte entries
1200 0859 CDB00C      call    addhla          ;compute address of entry
1201 085C D5          push    d
1202 085D 5E          mov     e,m          ;fetch entry
1203 085E 23          inx     h
1204 085F 56          mov     d,m
1205 0860 EB          xchg
1206 0861 D1          pop     d
1207 0862 E9          pchl          ;jump to it
1208                ;
1209 0863 434F4D20     typtbl: db     'COM '
1210 0867 53554220     db     'SUB '
1211 086B 50524C20     db     'PRL '
1212 086F 00          db     0
1213                ;
1214 0870 8908         xptbl: dw     xcom
1215 0872 CD07         dw     xsub
1216 0874 8908         dw     xcom
1217
1218
1219                ;
1220                ; unsuccessful attempt to open command file
1221                ;
1222 0876 C1          exec3: pop     b          ;recover drive
1223 0877 79          mov     a,c
1224 0878 12          stax    d          ;replace in fcb
1225 0879 C9          ret
1226                ;
1227                ;
1228                ; settype:
1229                ; set file type specified from type table
1230                ; a = offset (x2) of desired type (in bytes)
1231 087A 0F          rrc
1232 087B 216308      lxi     h,typtbl

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

1233 087E CDB00C      call    addh1a      ;h1 = type in type table
1234 0881 11B60D      lxi      d,fcbl+9
1235 0884 0E03        mvi      c,3
1236 0886 C3AE0B      jmp      move      ;move type into fcb
1237
1238
1239
1240
1241
1242 xcom:              ;DE = .fcb
1243
1244
1245
1246 0889 210001      lxi      h,tpa
1247 088C 22CE0D      shld     fcbr      ;set load address to 100h
1248 088F 2A9F0D      lhld     realdos-1 ;put fcb in the loader's stack
1249 0892 25          dcr      h          ;page below LOADER (or bottom RSX)
1250 0893 2ECO        mvi      l,0C0h     ;offset for FCB in page below the BDOS
1251 0895 E5          push     h          ;save for LOADER call
1252 0896 1A          ldax     d          ;get drive from fcb(0)
1253 0897 325000      sta      cmdrv      ;set command drive field in base page
1254 089A EB          xchg
1255 089B 0E23        mvi      c,35
1256 089D CDAE0B      call     move      ;now move FCB to the top of the TPA
1257
1258
1259
1260 08A0 21670D      lxi      h,errflg    ;tell parser to ignore errors
1261 08A3 34          inr      m
1262 08A4 2A6C0D      xcom3: lhld     parsep
1263 08A7 2B          dcx      h          ;backup over delimiter
1264 08A8 118100      lxi      d,buf+1
1265 08AB EB          xchg
1266 08AC 226C0D      shld     parsep      ;set parser to 81h
1267 08AF CDB70B      call     copy0      ;copy command tail to 81h with
1268                                     ;terminating 0 (returns A=length)
1269 08B2 328000      sta      buf          ;put command tail length at 80h
1270 08B5 CDB80A      xcom5: call     gfn      ;parse off first argument
1271 08B8 225100      shld     pass0
1272 08BB 7B          mov      a,b
1273 08BC 325300      sta      len0
1274 08BF 116C00      lxi      d,dfcb1
1275 08C2 CDB80A      call     gfn0      ;parse off second argument
1276 08C5 225400      shld     pass1
1277 08C8 7B          mov      a,b
1278 08C9 325600      sta      len1
1279 08CC 21710D      xcom7: lxi      h,chaindsk ;.CHAINDSK
1280 08CF 7E          mov      a,m
1281 08D0 B7          ora      a
1282 08D1 F48009      cp      select
1283 08D4 3A700D      lda      usernum
1284 08D7 CD8609      call     setuser      ;set default user, returns H=SCB
1285 08DA 87          add      a          ;shift user to high nibble
1286 08DB 87          add      a
1287 08DC 87          add      a
1288 08DD 87          add      a

```

COPYRIGHT © 1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

1289 08DE 2EDA      mvi    l,seldsk
1290 08E0 B6        ora     a           ;put disk in low nibble
1291 08E1 320400     sta     defdrv      ;set location 4
1292                ;
1293                ;       initialize stack
1294                ;
1295 08E4 D1          xcom8: pop     d           ;DE = ,fcb
1296 08E5 2A9F0D     lhld    realdos-1      ;base page of BDOS
1297 08E8 AF          xra     a
1298 08E9 6F          mov     l,a           ;top of stack below BDOS
1299 08EA F9          sphl                    ;change the stack pointer for CCP
1300 08EB 67          mov     h,a           ;push warm start address on stack
1301 08EC E5          push    h           ;for programs returning to the CCP
1302 08ED 24          inr     h           ;Loader will return to TPA
1303 08EE E5          push    h           ;after loading a transient program
1304                ;
1305                ;       initialize fcb0(CR), console mode, program return code
1306                ;       & removable media open and login vectors
1307                ;
1308 08EF 327C00     xcom9: sta     7ch           ;clear next record to read
1309 08F2 06CF       mvi     b,contmode
1310 08F4 CDF90B     call    setbyte          ;set to zero (turn off ^C status)
1311 08F7 2E90       mvi     l,olog
1312 08F9 77         mov     m,a           ;zero removable open login vector
1313 08FA 23         inx     h
1314 08FB 77         mov     m,a
1315 08FC 23         inx     h
1316 08FD 77         mov     m,a           ;zero removable media login vector
1317 08FE 23         inx     h
1318 08FF 77         mov     m,a
1319 0900 2EB3       mvi     l,ccpflagl
1320 0902 7E         mov     a,m
1321 0903 E680       ani     chain$flg      ;chaining?
1322 0905 C20D09     jnz     loader          ;load program without clearing
1323 0908 2EAC       mvi     l,prog$ret$code ;the program return code
1324 090A 77         mov     m,a           ;A=0
1325 090B 23         inx     h
1326 090C 77         mov     m,a           ;set program return = 0000h
1327                ;
1328                ;       call loader
1329                ;
1330 loader:
1331 090D 7E         mov     a,m           ;reset chain flag if set,
1332 090E E63F       ani     not$chainflg     ;has no effect if we fell through
1333 0910 77         mov     m,a
1334 0911 0E3B       mvi     c,loadf        ;use load RSX to load file
1335 0913 C30500     jmp     bdos           ;now load it
1336                ;
1337                ;
1338                ;
1339                ;
1340                ;*****
1341                ;
1342                ;       BDOS FUNCTION INTERFACE - Non FCB functions
1343                ;
1344                ;*****

```

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #


```

1345 ;
1346 ;
1347 ;
1348 ;;-----
1349 ;;
1350 ;;
1351 ;;
1352 ;; print character on terminal
1353 ;; pause if screen is full
1354 ;; (BDOS function #2)
1355 ;;
1356 ;; entry: a = character (putc entry)
1357 ;;         e = character (putc2 entry)
1358 ;;
1359
1360 0916 FE0A putc: cpi    lf          ;end of line?
1361 0918 C23B09 jnz     putc1       ;jump if not
1362 091B 21980D lxi     h,pgsize    ;pgsize
1363 091E 7E     mov     a,m      ;check page size
1364 091F 23     inx     h        ;.line
1365 0920 34     inr     a        ;line=line+1
1366 0921 96     sub     a        ;line=page?
1367 0922 C23909 jnz     putc0
1368 0925 77     mov     m,a      ;reset line=0 if so
1369 0926 23     inx     h        ;pgmode
1370 0927 7E     mov     a,m      ;is page mode off?
1371 0928 B7     ora     a        ;page=0 if so
1372 0929 11240D lxi     d,more
1373 092C CC4109 cz       getc      ;wait for input if page mode on
1374 092F FE03 cpi     ctrlc
1375 0931 CA0605 jz      ccpr
1376 0934 1E0D mvi     e,cr
1377 0936 CD3C09 call    putc2       ;print a cr
1378 0939 3E0A putc0: mvi    a,lf      ;print the end of line char
1379 093B 5F     putc1: mov     a,a
1380 093C 0E02 putc2: mvi    c,coutf
1381 093E C30500 jmp     bdos
1382
1383 ;;
1384 ;;-----
1385 ;;
1386 ;; get character from console
1387 ;; (BDOS function #1)
1388 ;;
1389 0941 CD4909 getc: call    pmsg
1390 0944 0E01 getc1: mvi    c,cinf
1391 0946 C30500 jmp     bdos
1392
1393 ;;-----
1394 ;;
1395 ;; print message string on terminal
1396 ;; (BDOS function #9)
1397 ;;
1398 0949 0E09 pmsg: mvi    c,pbuff
1399 094B C30500 jmp     bdos
1400

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

1401      ;; -----
1402      ;;
1403      ;;   read line from console
1404      ;;   (calls BDOS function #10)
1405      ;;
1406      ;;   exit:  z = set if null line
1407      ;;
1408      ;;   This function uses the buffer 'cbuf' (see definition of
1409      ;;   function 10 for a description of the buffer). All input
1410      ;;   is converted to upper case after reading and the pointer
1411      ;;   'parsep' is set to the beginning of the first non-white
1412      ;;   character string.
1413      ;;
1414      094E 21F40D  rcln: lxi    h,cbufmx    ;get line from terminal
1415      0951 36E7    mvi    m,comlen    ;set maximum buffer size
1416      0953 EB      xchg
1417      0954 0E0A    mvi    c,rbuff
1418      0956 CD0500  call    bdos
1419      0959 21F50D  lxi    h,cbufl    ;terminate line with zero byte
1420      095C 7E      mov    a,m
1421      095D 23      inx    h
1422      095E CDB00C  call    addhla
1423      0961 3600    mvi    a,0        ;put zero at the end
1424      0963 C3090C  jmp     crlf      ;advance to next line
1425      ;
1426      ;;
1427      ;; -----
1428      ;;
1429      ;;   exit routine if keyboard struck
1430      ;;   (calls BDOS function #11)
1431      ;;
1432      ;;   Control is returned to the caller unless the console
1433      ;;   keyboard has a character ready, in which case control
1434      ;;   is transfer to the main program of the CCP.
1435      ;;
1436      0966 CD6D09  break: call    break1
1437      0969 C8      rz
1438      096A C30605  jmp     ccpr
1439
1440      096D 0E0B    break1: mvi    c,cstatf
1441      096F CDEB09  call    rw
1442      0972 C8      rz
1443      0973 0E01    mvi    c,cinf
1444      0975 C3EB09  jmp     rw
1445
1446      ;;
1447      ;; -----
1448      ;;
1449      ;;
1450      ;;   set disk buffer address
1451      ;;   (BDOS function #26)
1452      ;;
1453      ;;   entry:  de -> buffer ('setbuf' only)
1454      ;;
1455      0978 118000  sbuf80: lxi    d,buf
1456      097B 0E1A    setbuf: mvi    c,dmaof

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

1457 097D C30500      jmp      bdos
1458                  ;;
1459                  ;;-----
1460                  ;;
1461                  ;;      select disk
1462                  ;;      (BDOS function #14)
1463                  ;;
1464                  ;;      entry:  a = drive
1465                  ;;
1466                  select:
1467 0980 5F            mov      e,a
1468 0981 0E0E          mvi      c,self
1469 0983 C30500      jmp      bdos
1470                  ;
1471                  ;;
1472                  ;;-----
1473                  ;;
1474                  ;;      set user number
1475                  ;;      (BDOS function #32)
1476                  ;;
1477                  ;;      entry:  a = user #
1478                  ;;      exit:   H = SCB page
1479                  ;;
1480                  setuser:
1481 0986 06E0          mvi      b,usrcode
1482 0988 C3F90B      jmp      set$byte
1483                  ;
1484                  ;
1485                  ;
1486                  ;*****
1487                  ;
1488                  ;      BDOS FUNCTION INTERFACE - Functions with a FCB Parameter
1489                  ;
1490                  ;*****
1491                  ;
1492                  ;
1493                  ;;
1494                  ;;      open file
1495                  ;;      (BDOS function #15)
1496                  ;;
1497                  ;;      exit:   z = set if file not found
1498                  ;;
1499                  ;;
1500                  opencom:      ;open command file (SUB, COM or PRL)
1501 098B 010F00      lxi      b,openf      ;b=0 => return error mode of 0
1502 098E 11AD0B      lxi      d,fcb        ;use internal FCB
1503                  ;
1504                  ;;      BDOS CALL ENTRY POINT      (used by built-ins)
1505                  ;;
1506                  ;;      entry:  b = return error mode (must be 0 or 0ffh)
1507                  ;;              c = function no.
1508                  ;;              de = .fcb
1509                  ;;      exit:   z = set if error
1510                  ;;              de = .fcb
1511                  ;;
1512 0991 212000      bdosf: lxi      h,32      ;offset to current record

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

1513 0994 19      dad    d          ;HL = .current record
1514 0995 3600    mvi    m,0        ;set to zero for read/write
1515 0997 C5      push   b          ;save function(C) & error mode(B)
1516 0998 D5      push   d          ;save .fcb
1517 0999 1A      ldax   d          ;was a disk specified?
1518 099A A0      ana    b          ;and with 0 or 0ffh
1519 099B 3D      dcr    a          ;if so, select it in case
1520 099C F48009  cp      select    ;of permanent error (if errmode = 0ffh)
1521 099F 11A20D  lxi    d,passwd
1522 09A2 CD7B09  call   setbuf    ;set dma to password
1523 09A5 D1      pop     d          ;restore .fcb
1524 09A6 C1      pop     b          ;restore function(C) & error mode(B)
1525 09A7 D5      push   d
1526 09A8 2A8D03  lhld   scbaddr
1527 09AB 2EE7    mvi    l,errormode
1528 09AD 70      mov     m,b        ;set error mode
1529 09AE E5      push   h          ;save .errormode
1530 09AF CD0500  call   bdos
1531 09B2 D1      pop     d          ;.errormode
1532 09B3 AF      xra    a
1533 09B4 12      stax   d          ;reset error mode to 0
1534 09B5 3A720D  lda     disk
1535 09B8 1EDA    mvi    e,seldsk
1536 09BA 12      stax   d          ;reset current disk to default
1537 09BB E5      push   h          ;save bdos return values
1538 09BC CD7809  call   sbuf80
1539 09BF E1      pop     h          ;bdos return
1540 09C0 2C      inr     l          ;set z flag if error
1541 09C1 D1      pop     d          ;restore .fcb
1542 09C2 C9      ret

```

```

1543 ;;
1544 ;;
1545 ;;
1546 ;;      close file
1547 ;;      (BDOS function #16)
1548 ;;
1549 ;;      exit:  z = set if close error
1550 ;;

```

```

1551 ;;close:  mvi    c,closef
1552 ;;        jmp    oc
1553 ;;

```

```

1554 ;;-----
1555 ;;
1556 ;;      delete file
1557 ;;
1558 ;;      exit:  z = set if file not found
1559 ;;

```

```

1560 ;;      The match any character '?' may be used without restriction
1561 ;;      for this function. All matched files will be deleted.
1562 ;;

```

```

1563 ;;
1564 ;;delete:
1565 ;;      mvi    c,delf
1566 ;;      jmp    oc
1567 ;;
1568 ;;-----

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

1569      ;;
1570      ;;      create file
1571      ;;      (BDOS function #22)
1572      ;;
1573      ;;      exit:  z = set if create error
1574      ;;
1575      ;;make:      mvi      c,makef
1576      ;;              jmp      oc
1577      ;;-----
1578      ;;
1579      ;;      search for first filename match (using 'DFCB' and 'BUF')
1580      ;;      (BDOS function #17)
1581      ;;
1582      ;;      exit:  z = set if no match found
1583      ;;              z = clear if match found
1584      ;;              de -> directory entry in buffer
1585      ;;
1586      09C3 0E11      srchf: mvi      c,searf      ;set search first function
1587      09C5 C3CA09      jmp      srch
1588      ;;
1589      ;;-----
1590      ;;
1591      ;;      search for next filename match (using 'DFCB' and 'BUF')
1592      ;;      (BDOS function #18)
1593      ;;
1594      ;;      exit:  z = set if no match found
1595      ;;              z = clear if match found
1596      ;;              de -> directory entry in buffer
1597      ;;
1598      09C8 0E12      srchn: mvi      c,searxf     ;set search next function
1599      09CA 115C00      srch: lxi      d,dfcb      ;use default fcb
1600      09CD CD0500      call     bdos
1601      09D0 3C              inr      a              ;return if not found
1602      09D1 C8              rz
1603      09D2 3D              dcr      a              ;restore original return value
1604      09D3 87              add      a              ;shift to compute buffer pos'n
1605      09D4 87              add      a
1606      09D5 87              add      a
1607      09D6 87              add      a
1608      09D7 87              add      a
1609      09DB 218000      lxi      h,buf          ;add to buffer start address
1610      09DB CDB00C      call     addhla
1611      09DE EB              xchg
1612      09DF AF              xra      a              ;de -> entry in buffer
1613      09E0 3D              dcr      a              ;may be needed to clear z flag
1614      09E1 C9              ret              ;depending of value of 'buf'
1615      ;;
1616      ;;-----
1617      ;;
1618      ;;      read file
1619      ;;      (BDOS function #20)
1620      ;;
1621      ;;      entry: hl = buffer address (readb only)
1622      ;;      exit:  z = set if read ok
1623      ;;
1624      09E2 AF      read: xra      a              ;clear getc pointer

```

COPYRIGHT © 1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. #

```

1625 09E3 329F0D      sta      bufp
1626 09E6 0E14        mvi      c,readf
1627 09EB 115C00      lxi      d,dfcb
1628 09EB CD0500      rw:     call    bdos
1629 09EE B7          ora      a
1630 09EF C9          ret
1631 ;
1632 ;
1633 ;-----
1634 ;
1635 ;   $$$SUB interface
1636 ;
1637 ;   entry: c = bdos function number
1638 ;   exit  z = set if successful
1639 ;
1640 09F0 11730D      sudos: lxi      d,subfcb
1641 09F3 C3EB09      jmp      rw
1642 ;
1643 ;
1644 ;
1645 ;*****
1646 ;
1647 ;   COMMAND LINE PARSING SUBROUTINES
1648 ;
1649 ;*****
1650 ;
1651 ;-----
1652 ;
1653 ;   COMMAND LINE PREPARSER
1654 ;   reset function 10 flag
1655 ;   set up parser
1656 ;   convert to upper case
1657 ;
1658 ;   All input is converted to upper case and the pointer
1659 ;   'parsep' is set to the begining of the first non-blank
1660 ;   character string. If the line begins with a ; or !, it
1661 ;   is treated specially:
1662 ;
1663 ;           ;   comment      the line is ignored
1664 ;           :   conditional  the line is ignored if a fatal
1665 ;                           error occured during the previous
1666 ;                           command, otherwise the ! is
1667 ;                           ignored
1668 ;
1669 ;   An exclamation point is used to separate multiple commands on a
1670 ;   a line. Two adjacent exclamation points translates into a single
1671 ;   exclamation point in the command tail for compatibility.
1672 ;-----
1673 ;
1674 ;
1675 uc:
1676 09F6 CDEE0B      call    resetccpf1g
1677 09F9 EB          xchg                    ;DE = .SCB
1678 09FA AF          xra      a
1679 09FB 32A10D      sta      option          ;zero option flag
1680 09FE 21F60D      lxi      h,cbuf

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

1681 0A01 CD440A      call    skps1      ;skip leading spaces/tabs
1682 0A04 EB          xchg
1683 0A05 FE3B        cpi      '/'      ;HL = ,scb
1684 0A07 C8          rz
1685 0A08 FE21        cpi      '!'
1686 0A0A CA1C0A      jz       uc0
1687 0A0D FE3A        cpi      '/'
1688 0A0F C21D0A      jnz      uc1
1689 0A12 2EAC        mvi      l,prog$reticode
1690 0A14 34          inr       n
1691 0A15 34          inr       n      ;was ^C typed? (low byte 0FEh)
1692 0A16 CA1C0A      jz       uc0      ;successful, if so
1693 0A19 23          inx       h
1694 0A1A 34          inr       n      ;is high byte 0FFh?
1695 0A1B C8          rz          ;skip command, if so
1696 0A1C 13          uc0: inx     d      ;skip over 1st character
1697 0A1D EB          uc1: xchg
1698 0A1E 226C0D      shld     parsep ;set parse pointer to beginning of line
1699 0A21 7E          uc3: mov     a,n    ;convert lower case to upper
1700 0A22 FE5B        cpi      'l'
1701 0A24 C22A0A      jnz      uc4
1702 0A27 32A10D      sta      option ;'l' is the option delimiter => command option
1703 0A2A FE61        uc4: cpi      'a'
1704 0A2C DA370A      jc       uc5
1705 0A2F FE7B        cpi      'z'+1
1706 0A31 D2370A      jnc      uc5
1707 0A34 D620        sui      'a'-'A'
1708 0A36 77          mov     n,a
1709                uc5:
1710                if multi
1711 0A37 FE21        cpi      '!'
1712 0A39 CC590A      cz       multistart ;HL=,char, A=char
1713                endif
1714 0A3C 23          inx       h      ;advance to next character
1715 0A3D B7          ora      a      ;loop if not end of line
1716 0A3E C2210A      jnz      uc3
1717                ;
1718                ;      skip spaces
1719                ;      return with zero flag set if end of line
1720                ;
1721 0A41 2A6C0D      skps: lhd     parsep ;get current position
1722 0A44 226C0D      skps1: shld    parsep ;save position
1723 0A47 229B0D      shld     errorp ;save position for error message
1724 0A4A 7E          mov     a,n
1725 0A4B B7          ora      a      ;return if end of command
1726 0A4C C8          rz
1727 0A4D FE20        cpi      ' '
1728 0A4F CA550A      jz       skps2
1729 0A52 FE09        cpi      tab    ;skip spaces & tabs
1730 0A54 C0          rnz
1731 0A55 23          skps2: inx     h      ;advance past space/tab
1732 0A56 C3440A      jmp      skps1 ;loop
1733                ;
1734                ;-----
1735                ;
1736                ;      MULTIPLE COMMANDS PER LINE HANDLER

```

CP/M - PLUS V3.0

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135

```

1737 ;
1738 ;-----
1739 if multi
1740
1741 multistart:
1742 ;
1743 ; A = current character in command line
1744 ; HL = address of current character in command line
1745 ;
1746 ;double exclamation points become one
1747 0A59 5D mov e,l
1748 0A5A 54 mov d,h
1749 0A5B 13 inx d
1750 0A5C 1A ldax d
1751 0A5D FE21 cpi '!' ;double exclamation points
1752 0A5F F5 push psf
1753 0A60 E5 push h
1754 0A61 CCB70B cz copy0 ;convert to one, if so
1755 0A64 E1 pop h
1756 0A65 F1 pop psf
1757 0A66 C8 rz
1758 ;we have a valid multiple command line
1759 0A67 3600 mvi a,0 ;terminate command line here
1760 0A69 EB xchg
1761 ;multiple commands not allowed in submits
1762 ;NOTE: submit unravels multiple commands making the
1763 ;following test unnecessary. However, with GET[system]
1764 ;or CP/M 2.2 SUBMIT multiple commands will be postponed
1765 ;until the entire submit completes...
1766 ; call subtest ;submit active
1767 ; mvi a,0
1768 ; rnz ;return with A=0, if so
1769 ;set up the RSX buffer
1770 0A6A 2A0600 lhd osbase ;get high byte of TPA address
1771 0A6D 25 dcr h ;subtract 1 page for buffer
1772 0A6E 2E18 mvi l,endchain ;HL = RSX buffer base-1
1773 0A70 77 mov a,a ;set end of chain flag to 0
1774 0A71 E5 push h ;save it
1775 0A72 23 multi0: inx h
1776 0A73 13 inx d
1777 0A74 1A ldax d ;get character from cbuf
1778 0A75 77 mov a,a ;place in RSX
1779 0A76 FE21 cpi '!'
1780 0A78 C27D0A jnz multil
1781 0A7B 360D mvi a,cr ;change exclamation point to cr
1782 0A7D B7 multil: ora a
1783 0A7E C2720A jnz multi0
1784 0A81 360D mvi a,cr ;end last command with cr
1785 0A83 23 inx h
1786 0A84 77 mov a,a ;terminate with a zero
1787 ;set up RSX prefix
1788 0A85 2E06 mvi l,6 ;entry point
1789 0A87 36C3 mvi a,jmp ;put a jump instruction there
1790 0A89 23 inx h
1791 0A8A 3609 mvi a,9 ;make it a jump to base+9 (RSX exit)
1792 0A8C 23 inx h

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #


```

1793 0A8D 74      mov     m,h
1794 0ABE 23      inx     h           ;HL = RSX exit point
1795 0ABF 36C3     mvi     m,jmp       ;put a jump instruction there
1796 0A91 2E0E     mvi     l,warmflg    ;HL = remove on warm start flag
1797 0A93 77      mov     m,a           ;set (0) for RSX to remain resident
1798 0A94 6F      mov     l,a           ;set low byte to 0 for fixchain
1799 0A95 EB      xchg     ;DE = RSX base
1800 0A96 CDD001   call    fixchain    ;add the RSX to the chain
1801             ;save buffer address
1802 0A99 2ABD03   lhld    scbaddr
1803 0A9C 2EB1     mvi     l,ccpconbuf    ;save buffer address in CCP conbuf field
1804 0A9E D1      pop     d           ;DE = RSX base
1805 0A9F 13      inx     d
1806 0AA0 73      mov     m,e
1807 0AA1 23      inx     h
1808 0AA2 72      mov     m,d
1809 0AA3 2EAE     mvi     l,multi$rsx$pg
1810 0AA5 72      mov     m,d           ;save the RSX base
1811 0AA6 AF      xra     a           ;zero in a to fall out of uc
1812 0AA7 C9      ret
1813             ;
1814             ;
1815             ;      save the BDOS conbuffer address and
1816             ;      terminate RSX if necessary.
1817             ;
1818             ;      multisave!
1819 0AA8 11B1BA     lxi     d,conbuffer*256+ccpconbuf
1820 0AAB CDA70B     call   wordmov       ;first copy conbuffer in case SUBMIT
1821 0AAE B7      ora     a           ;and/or GET are active
1822 0AAF 11B1BC     lxi     d,conbuff1*256+ccpconbuf
1823 0AB2 CCA70B     cz      wordmov       ;if conbuff is zero then conbuff1 has the
1824 0AB5 E5      push    h           ;next address
1825 0AB6 CD6D09     call   break1
1826 0AB9 E1      pop     h           ;H = SCB page
1827 0ABA 2EB1     mvi     l,ccpconbuf
1828 0ABC C2CB0A     jnz    multiend
1829 0ABF 5E      mov     e,a
1830 0AC0 23      inx     h
1831 0AC1 56      mov     d,a           ;DE = next conbuffer address
1832 0AC2 34      inr     m
1833 0AC3 35      dcr     m           ;is high byte zero?
1834 0AC4 2B      dcx     h           ;HL = .ccpconbuf
1835 0AC5 CACB0A     jz     multiend    ;remove multicaud RSX if so
1836 0AC8 1A      ldax    d           ;check for terminating zero
1837 0AC9 B7      ora     a
1838 0ACA C0      rnz     ;return if not
1839             ;
1840             ;      we have exhausted all the commands
1841             ;      multiend!
1842             ;      HL = .ccpconbuf
1843 0ACB AF      xra     a
1844 0ACC 77      mov     m,a           ;set buffer to zero
1845 0ACD 23      inx     h
1846 0ACE 77      mov     m,a
1847 0ACF 2EAE     mvi     l,multi$rsx$pg
1848 0AD1 66      mov     h,m

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

1849 0AD2 2E0E      mvi    1,0eh      ;HL=RSX remove on warmstart flag
1850 0AD4 35        dcr     m          ;set to true for removal
1851 0AD5 C30002    jmp     rsx$chain  ;remove the multicmd rsx buffer
1852
1853                endif
1854                ;;
1855                ;*****
1856                ;
1857                ;   FILE NAME PARSER
1858                ;
1859                ;*****
1860                ;
1861                ;
1862                ;
1863                ;   get file name (read in if none present)
1864                ;
1865                ;
1866                ;;   The file-name parser in this CCP implements
1867                ;;   a user/drive specification as an extension of the normal
1868                ;;   CP/M drive selection feature. The syntax of the
1869                ;;   user/drive specification is given below. Note that a
1870                ;;   colon must follow the user/drive specification.
1871                ;;
1872                ;;   <a>:   <a> is an alphabetic character A-P specifying one
1873                ;;           of the CP/M disk drives.
1874                ;;
1875                ;;   <n>:   <n> is a decimal number 0-15 specifying one of the
1876                ;;           user areas.
1877                ;;
1878                ;;   <n><a>: A specification of both user area and drive.
1879                ;;
1880                ;;   <a><n>: Synonymous with above.
1881                ;;
1882                ;;   Note that the user specification cannot be included
1883                ;;   in the parameters of transient programs or precede a file
1884                ;;   name. The above syntax is parsed by gcmd (get command).
1885                ;;
1886                ;; *****
1887
1888                getfn:
1889                    if prompts
1890                        lxi     d,fmsg
1891                getfn0:
1892                    call     getprm
1893                    endif
1894 0AD8 115C00      gfn:  lxi     d,dfcb
1895 0ADB CD410A      gfn0: call     skps      ;sets zero flag if eol
1896 0ADE F5          push     psw
1897 0ADF CDE40A      call     gfn2
1898 0AE2 F1          pop      psw
1899 0AE3 C9          ret
1900                ;
1901                ;   BDOS FUNCTION 152 INTERFACE
1902                ;
1903                ;entry: DE = ,FCB
1904                ;       HL = ,buffer

```

COPYRIGHT © 1982
DIGITAL RESEARCH

P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

1905          ;flags/A reg preserved
1906          ;exit: DE = ,FCB
1907          ;
1908          ;
1909 0AE4 226C0D gfn2: shld parsep
1910 0AE7 22980D      shld errorp
1911 0AEA D5         push d          ;save .fcb
1912 0AEB 116C0D      lxi d,pfncb
1913 0AEE 0E9B        mvi c,parsef
1914          if func152
1915 0AF0 CD0500      call bdos
1916          else
1917          call parse
1918          endif
1919 0AF3 D1          pop d          ;.fcb
1920 0AF4 7C          mov a,h
1921 0AF5 B5          ora l          ;end of command? (HL = 0)
1922 0AF6 46          mov b,m        ;get delimiter
1923 0AF7 23          inx h          ;move past delimiter
1924 0AF8 C2FE0A      jnz gfn3
1925 0AFB 21280B      lxi h,zero+2   ;set HL = .0
1926 0AFE 7C          gfn3: mov a,h
1927 0AFF B5          ora l          ;parse error? (HL = 0ffffh)
1928 0B00 C2090B      jnz gfn4
1929 0B03 21280B      lxi h,zero+2
1930 0B06 CD3A0C      call perror
1931 0B09 7B          gfn4: mov a,b
1932 0B0A FE2E        cpi ','
1933 0B0C C2100B      jnz gfn6
1934 0B0F 2B          dcx h
1935 0B10 226C0D      gfn6: shld parsep ;update parse pointer
1936 0B13 0E10      gfnpwd: mvi c,16
1937 0B15 21D00D      lxi h,pfcb
1938 0B18 D5         push d
1939 0B19 CD4E0B      call move
1940 0B1C 11A20D      lxi d,passwd   ;HL = .disk map in pfcb
1941 0B1F 0E0A        mvi c,10
1942 0B21 CD4E0B      call move      ;copy to passwd
1943 0B24 D1          pop d          ;HL = .password len
1944 0B25 7E          mov a,m
1945 0B26 210000      zero: lxi h,0   ;must be an "lxi h,0"
1946 0B29 B7          ora a          ;is there a password?
1947 0B2A 47          mov b,a
1948 0B2B CA380B      jz gfn8
1949 0B2E 2A9B0D      lhld errorp    ;HL = .filename
1950 0B31 7E          gfn7: mov a,m
1951 0B32 FE3B        cpi ';'
1952 0B34 23          inx h
1953 0B35 C2310B      jnz gfn7
1954 0B38 C9          gfn8: ret       ;B = len, HL = .password
1955
1956          ;
1957          ; PARSE CP/M 3 COMMAND
1958          ; entry: DE = ,UFCB (user no. byte in front of FCB)
1959          ; PARSEP = ,command line
1960          gcnd:

```

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

1961 0B39 DE      push    d
1962 0B3A AF      xra     a
1963 0B3B 12      stax    d          ;clear user byte
1964 0B3C 13      inx     d
1965 0B3D 12      stax    d          ;clear drive byte
1966 0B3E 13      inx     d
1967 0B3F CD410A  call    skps      ;skip leading spaces
1968                ;
1969                ;      Begin by looking for user/drive-spec. If none is found,
1970                ;      fall through to main file-name parsing section. If one is found
1971                ;      then branch to the section that handles them. If an error occurs
1972                ;      in the user/drive spec; treat it as a filename for compatibility
1973                ;      with CP/M 2.2. (e.g. STAT VAL; etc.)
1974                ;
1975 0B42 2A6C0D     lhld    parsep      ;get pointer to current parser position
1976 0B45 D1        pop     d
1977 0B46 D0        push    d          ;DE = ,UFCB
1978 0B47 0604     mvi     b,4        ;maximum length of user/drive spec
1979 0B49 7E      gcmd1: mov     a,m      ;get byte
1980 0B4A FE3A     cpi     ':'        ;end of user/drive-spec?
1981 0B4C CA670B     jz      gcmd2      ;parse user/drive if so
1982 0B4F B7       ora     a          ;end of command?
1983 0B50 CA580B     jz      gcmd8      ;parse filename (Func 152), if so
1984 0B53 05       dcr     b          ;maximum user/drive spec length exceeded?
1985 0B54 23       inx     h
1986 0B55 C2490B     jnz     gcmd1      ;loop if not
1987                ;
1988                ;      Parse filename, type and password
1989                ;
1990      gcmd8:
1991 0B5B D1        pop     d
1992 0B59 AF      xra     a
1993 0B5A 12      stax    d          ;set user = default
1994 0B5B 2A6C0D     lhld    parsep
1995 0B5E 13      gcmd9: inx     d      ;past user number byte
1996 0B5F 1A      ldax    d          ;A=drive
1997 0B60 F5      push    psw
1998 0B61 CDE40A  call    gfn2      ;BDOS function 152 interface
1999 0B64 F1      pop     psw
2000 0B65 12      stax    d
2001 0B66 C9      ret
2002                ;
2003                ;      Parse the user/drive-spec
2004                ;
2005      gcmd2:
2006 0B67 2A6C0D     lhld    parsep      ;get pointer to beginning of spec
2007 0B6A 7E      mov     a,m      ;get character
2008 0B6B FE30     gcmd3: cpi     '0'    ;check for user number
2009 0B6D DA850B     jc      gcmd4      ;jump if not numeric
2010 0B70 FE3A     cpi     '9'+1
2011 0B72 D2850B     jnc     gcmd4
2012 0B75 CD710C  call    gdns      ;get the user # (returned in B)
2013 0B78 D1      pop     d
2014 0B79 D5      push    d
2015 0B7A 1A      ldax    d          ;see if we already have a user #
2016 0B7B B7      ora     a

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

2017 0B7C C2580B      jnz    gcmd8      ;skip if we do
2018 0B7F 78          mov     a,b        ;A = specified user number
2019 0B80 3C          inr     a          ;save it as the user-spec
2020 0B81 12          stax    d
2021 0B82 C39C0B      jmp     gcmd5
2022 0B85 FE41      gcmd4: cpi     'A'      ;check for drive-spec
2023 0B87 DA580B      jc       gcmd8      ;skip if not a valid drive character
2024 0B8A FE51          cpi     'P'+1
2025 0B8C D2580B      jnc     gcmd8
2026 0B8F D1          pop     d
2027 0B90 D5          push    d
2028 0B91 13          inx     d
2029 0B92 1A          ldax    d          ;see if we already have a drive
2030 0B93 B7          ora     a
2031 0B94 C2580B      jnz     gcmd8      ;skip if so
2032 0B97 7E          mov     a,m
2033 0B98 D640          sui     'Q'      ;convert to a drive-spec
2034 0B9A 12          stax    d
2035 0B9B 23          inx     h
2036 0B9C 7E      gcmd5: mov     a,m      ;get next character
2037 0B9D FE3A          cpi     ':'      ;end of user/drive-spec?
2038 0B9F C26B0B      jnz     gcmd3      ;loop if not
2039 0BA2 23          inx     h
2040 0BA3 D1          pop     d          ;.ufcb
2041 0BA4 C35E0B      jmp     gcmd9      ;parse the file name

```

2042

2043

2044

2045

2046

2047

2048

2049

2050

2051

2052

2053

2054

2055

2056

2057

2058

2059

2060

2061

2062

2063

2064

2065

2066

2067

2068

2069

2070

2071

2072

```

;
;*****
;
;          TEMPORARY PARSE CODE
;
;*****
if not func152
;    version 3.0b  Oct 08 1982 - Doug Huskey
;
;
passwords      equ      true

parse: ; DE->.(.filename,.fcb)
;
; filename = [d:]file[,type][;password]
;
; fcb assignments
;
; 0    => drive, 0 = default, 1 = A, 2 = B, ...
; 1-8  => file, converted to upper case,
;        padded with blanks (left justified)
; 9-11 => type, converted to upper case,
;        padded with blanks (left justified)
; 12-15 => set to zero
; 16-23 => password, converted to upper case,
;        padded with blanks
; 26   => length of password (0 - 8)

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```

2073      ;
2074      ; Upon return, HL is set to FFFFH if DE locates
2075      ;         an invalid file name;
2076      ; otherwise, HL is set to 0000H if the delimiter
2077      ;         following the file name is a 00H (NULL)
2078      ;         or a 0DH (CR);
2079      ; otherwise, HL is set to the address of the delimiter
2080      ;         following the file name.
2081      ;
2082      xchg
2083      mov     e,m           ;get first parameter
2084      inc     h
2085      mov     d,m
2086      push    d             ;save .filename
2087      inc     h
2088      mov     e,m           ;get second parameter
2089      inc     h
2090      mov     d,m
2091      pop     h             ;DE=.fcb HL=.filename
2092      xchg
2093      parse0:
2094      push    h             ;save .fcb
2095      xra     a
2096      mov     m,a           ;clear drive byte
2097      inc     h
2098      lxi     b,20h*256+11
2099      call    pad           ;pad name and type w/ blanks
2100      lxi     b,4
2101      call    pad           ;EXT, S1, S2, RC = 0
2102      lxi     b,20h*256+8
2103      call    pad           ;pad password field w/ blanks
2104      lxi     b,12
2105      call    pad
2106      call    skip
2107      ;
2108      ; check for drive
2109      ;
2110      ldax    d
2111      cpi     ':'           ;is this a drive?
2112      dcx     d
2113      pop     h
2114      push    h             ;HL = .fcb
2115      jnz     parsetname
2116      ;
2117      ; Parse the drive-spec
2118      ;
2119      parsedrv:
2120      ldax    d             ;get character
2121      ani     5fh           ;convert to upper case
2122      sui     'A'
2123      jc      perri
2124      cpi     16
2125      jnc     perri
2126      inc     d
2127      inc     d             ;past the ':'
2128      inr     a             ;set drive relative to 1

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2129      mvi     a,a           ;store the drive in FCB(0)
2130      ;
2131      ;   Parse the file-name
2132      ;
2133      parse$name:
2134          inx     h           ;HL = .fcb(1)
2135          call    delim
2136          jz      parse$ok
2137      if passwords
2138          lxi     b,7*256
2139      else
2140          mvi     b,7
2141      endif
2142      parse6:      ldax    d           ;get a character
2143          cpi     ','          ;file-type next?
2144          jz      parse$type    ;branch to file-type processing
2145          cpi     ';'
2146          jz      parse$pw
2147          call    gfc           ;process one character
2148          jnz     parse6        ;loop if not end of name
2149          jmp     parse$ok
2150      ;
2151      ;   Parse the file-type
2152      ;
2153      parse$type:
2154          inx     d           ;advance past dot
2155          pop     h
2156          push    h           ;HL = .fcb
2157          lxi     b,9
2158          dad     b           ;HL = .fcb(9)
2159      if passwords
2160          lxi     b,2*256
2161      else
2162          mvi     b,2
2163      endif
2164      parse8:      ldax    d
2165          cpi     ';'
2166          jz      parse$pw
2167          call    gfc           ;process one character
2168          jnz     parse8        ;loop if not end of type
2169      ;
2170      parse$ok:
2171          pop     b
2172          push    d
2173          call    skip
2174          call    delim
2175          pop     h
2176          rnz
2177          lxi     a,0
2178          ora     a
2179          rz
2180          cpi     cr
2181          rz
2182          xchg
2183          ret
2184      ;

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2185      ;      handle parser error
2186      ;
2187      perr:
2188          pop      b                      ;throw away return addr
2189      perr1:
2190          pop      b
2191          lxi      h,0ffffh
2192          ret
2193      ;
2194      if passwords
2195      ;
2196      ;      Parse the password
2197      ;
2198      parsepw:
2199          inx      d
2200          pop      h
2201          push     h
2202          lxi      b,16
2203          dad      b
2204          lxi      b,7*256+1
2205      parsepw1:
2206          call     gfc
2207          jnz      parsepw1
2208          mvi      a,7
2209          sub      b
2210          pop      h
2211          push     h
2212          lxi      b,26
2213          dad      b
2214          mov      m,a
2215          ldax     d                      ;delimiter in A
2216          jmp      parse$ok
2217      else
2218      ;
2219      ;      skip over password
2220      ;
2221      parsepw:
2222          inx      d
2223          call     delim
2224          jnz      parsepw
2225          jmp      parse$ok
2226      endif
2227      ;
2228      ;      get next character of name, type or password
2229      ;
2230      gfc:  call     delim                ;check for end of filename
2231          rz                      ;return if so
2232          cpi      ' '                ;check for control characters
2233          inx      d
2234          jc      perr                ;error if control characters encountered
2235          inr      b                  ;error if too big for field
2236          dcr      b
2237          jm      perr
2238      if passwords
2239          inr      c
2240          dcr      c

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____


```

2241      jnz      gfc1
2242  endif
2243      cpi      '*'          ;trap 'match rest of field' character
2244      jz       setwild
2245  gfc1:  mov     m,a          ;put character in fcb
2246      inc     h
2247      dec     b              ;decrement field size counter
2248      ora     a              ;clear zero flag
2249      ret
2250  ;;
2251  setwild:
2252      mvi     m,'?'          ;set match one character
2253      inc     h
2254      dec     b
2255      jp      setwild
2256      ret
2257  ;
2258  ;      skip spaces
2259  ;
2260  skip0: inc     d
2261  skip:  ldax   d
2262      cpi     ' '            ;skip spaces & tabs
2263      jz      skip0
2264      cpi     tab
2265      jz      skip0
2266      ret
2267  ;
2268  ;      check for delimiter
2269  ;
2270  ;      entry:  A = character
2271  ;      exit:   z = set if char is a delimiter
2272  ;
2273  delimiters: db      cr,tab,'.,;[]=<>!',0
2274
2275  delim: ldax   d              ;get character
2276      push   h
2277      lxi    h,delimiters
2278  delim1: cmp    m              ;is char in table
2279      jz     delim2
2280      inc    m
2281      dec    m              ;end of table? (0)
2282      inc    h
2283      jnz    delim1
2284      ora    a              ;reset zero flag
2285  delim2: pop    h
2286      rz
2287  ;
2288  ;      not a delimiter, convert to upper case
2289  ;
2290      cpi    'a'
2291      rc
2292      cpi    'z'+1
2293      jnc    delim3
2294      ani    05fh
2295  delim3: ani    07fh
2296      ret              ;return with zero set if so

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2297      ;
2298      ;      pad with blanks
2299      ;
2300      pad:  mov     m,b
2301           inx     h
2302           dcr     c
2303           jnz     pad
2304           ret
2305      ;
2306      endif
2307      ;
2308      ;
2309      ;*****
2310      ;
2311      ;      SUBROUTINES
2312      ;
2313      ;*****
2314      ;
2315      ;      if multi
2316      ;
2317      ;      copy SCB memory word
2318      ;      d = source offset e = destination offset
2319      ;
2320      wordmov:
2321      OBA7 2ABD03      lhd     scbaddr
2322      OBAA 6A          mov     l,d
2323      OBAB 54          mov     d,h
2324      OBAC 0E02        mvi     c,2
2325      ;
2326      ;      endif
2327      ;
2328      ;      copy memory bytes
2329      ;      de = destination hl = source c = count
2330      ;
2331      move:
2332      OBAE 7E          mov     a,m
2333      OBAF 12          stax     d      ;move byte to destination
2334      OBB0 23          inx     h
2335      OBB1 13          inx     d      ;advance pointers
2336      OBB2 0D          dcr     c      ;loop if non-zero
2337      OBB3 C2AE0B      jnz     move
2338      OBB6 C9          ret
2339      ;
2340      ;      copy memory bytes with terminating zero
2341      ;      hl = destination de = source
2342      ;      returns c=length
2343      ;
2344      OBB7 0E00      copy0: mvi     c,0
2345      OBB9 1A          copy1: ldax     d
2346      OBBA 77          mov     m,a
2347      OBBC 87          ora     a
2348      OBBD 79          mov     a,c
2349      OBBD C8          rz
2350      OBBD 23          inx     h
2351      OBBD 13          inx     d
2352      OBBD 03          inx     b

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

2353 0BC1 C3B90B      jmp     copy1
2354
2355                ;;
2356                ;;-----
2357                ;;
2358                ;;   get byte from file
2359                ;;
2360                ;;   exit:  z = set if byte gotten
2361                ;;         a = byte read
2362                ;;         z = clear if error or eof
2363                ;;         a = return value of bdos read call
2364                ;;
2365 0BC4 AF      getb:  xra     a           ;clear accumulator
2366 0BC5 219F0D      lxi     h,bufp       ;advance buffer pointer
2367 0BC8 34      inr     m
2368 0BC9 FCE209      cm     read         ;read sector if buffer empty
2369 0BCC B7      ora     a
2370 0BCD C0      rnz
2371 0BCE 3A9F0D      lda     bufp         ;compute pointer into buffer
2372 0BD1 218000      lxi     h,buf
2373 0BD4 CDB00C      call    addhla
2374 0BD7 AF      xra     a           ;set zero flag
2375 0BD8 7E      mov     a,m         ;get byte
2376 0BD9 C9      ret
2377                ;;
2378                ;;-----
2379                ;;
2380                ;;
2381                ;;   system control block flag routines
2382                ;;
2383                ;;   entry: c = bit mask (1 bit on)
2384                ;;         b = scb byte offset
2385                ;;
2386                subtest:
2387 0BDA 0140B4      lxi     b,submit
2388                getflg:
2389                ;   return flag value
2390                ;   exit:  zero flag set if flag reset
2391                ;         c = bit mask
2392                ;         hl = flag byte address
2393                ;
2394 0BDD 2ABD03      lhld    scbaddr
2395 0BE0 68      mov     l,b
2396 0BE1 7E      mov     a,m
2397 0BE2 A1      ana     c           ; a = bit
2398 0BE3 C9      ret
2399                ;
2400                setccpflg:
2401 0BE4 01A0B4      lxi     b,ccp10
2402                ;
2403                ;
2404                setflg:
2405                ;   set flag on (bit = 1)
2406                ;
2407 0BE7 CDD00B      call    getflg
2408 0BEA 79      mov     a,c

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

2409 00B B6      ora    a
2410 0BEC 77      mov    m,d
2411 0BED C9      ret
2412             ;
2413             resetccpflg:
2414 0BEE 01A0B4   lxi    b,ccp10
2415             ;
2416             resetflg:
2417             ; reset flag off (bit = 0)
2418             ;
2419 0BF1 CDD0B     call   getflg
2420 0BF4 79      mov    a,c
2421 0BF5 2F      cma
2422 0BF6 A6      ana    a
2423 0BF7 77      mov    m,d
2424 0BF8 C9      ret
2425             ;;
2426             ;;
2427             ;; SET/GET SCB BYTE
2428             ;;
2429             ;; entry:  A = byte ("setbyte" only)
2430             ;;          B = SCB byte offset from page
2431             ;;
2432             ;; exit:   A = byte ("getbyte" only)
2433             ;;
2434             setbyte:
2435 0BF9 2A8D03   lhld   scbaddr
2436 0BFC 68      mov    l,b
2437 0BFD 77      mov    m,d
2438 0BFE C9      ret
2439             ;
2440             getbyte:
2441 0BFF 2A8D03   lhld   scbaddr
2442 0C02 68      mov    l,b
2443 0C03 7E      mov    a,m
2444 0C04 C9      ret
2445             ;
2446             ;
2447             ;
2448             ;
2449             ;;-----
2450             ;;
2451             ;;
2452             ;; print message followed by newline
2453             ;;
2454             ;; entry:  de -> message string
2455             ;;
2456 0C05 CD4909   pmsgnl: call   pmsg
2457             ;
2458             ; print crlf
2459             ;
2460 0C08 45      dirln: mov    b,l                ;number of columns for DIR
2461 0C09 3E0D     crlf:  mov    a,cr
2462 0C0B CDA60C   call   pfc
2463 0C0E 3E0A     mov    a,l
2464 0C10 C3A60C   jmp     pfc

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2465      ;;
2466      ;;-----
2467      ;;
2468      ;;      print decimal byte
2469      ;;
2470      OC13 D60A      pdb:   sui     10
2471      OC15 DA250C      jc      pdb2
2472      OC18 1E30      mvi     e,'0'
2473      OC1A 1C      pdb1:  inr     e
2474      OC1B D60A      sui     10
2475      OC1D D21A0C      jnc     pdb1
2476      OC20 F5      push    psw
2477      OC21 CD3C09      call    putc2
2478      OC24 F1      pop     psw
2479      OC25 C63A      pdb2:  adi     10+ '0'
2480      OC27 C31609      jmp     putc
2481      ;;-----
2482      ;;
2483      ;;
2484      ;;      print string terminated by 0 or char in c
2485      ;;
2486      OC2A 7E      pstrg:  mov     a,m          ;get character
2487      OC2B B7      ora     a
2488      OC2C C8      rz
2489      OC2D B9      cmp     c
2490      OC2E C8      rz
2491      OC2F CDA60C      call    pfc          ;print character
2492      OC32 23      inx     h          ;advance pointer
2493      OC33 C32A0C      jmp     pstrg        ;loop
2494      ;;
2495      ;;-----
2496      ;;
2497      ;;      check for end of command (error if extraneous parameters)
2498      ;;
2499      OC36 CD410A      eoc:   call    skps
2500      OC39 C8      rz
2501      ;
2502      ;      handle parser error
2503      ;
2504      perror:
2505      OC3A 21670D      lxi     h,errflg
2506      OC3D 7E      mov     a,m
2507      OC3E B7      ora     a          ;ignore error????
2508      OC3F 3600      mvi     a,0      ;clear error flag
2509      OC41 C0      rnz          ;yes...just return to CCPRET
2510      OC42 2A9B0D      lhld    errorp    ;get pointer to what we're parsing
2511      OC45 0E20      mvi     c,' '
2512      OC47 CD2A0C      call    pstrg
2513      OC4A 3E3F      perr2:  mvi     a,'?'    ;print question mark
2514      OC4C CD1609      call    putc
2515      OC4F C30605      jmp     ccpr
2516      ;
2517      ;;-----
2518      ;;
2519      ;;
2520      ;;      print error message and exit processor

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2521      ;;
2522      ;;      entry: bc -> error message
2523      ;;
2524      ;;msgerr:      push      b
2525      ;;      call      crlf
2526      ;;      pop       d
2527      ;;      jmp       msgnl
2528      ;;
2529      ;;-----
2530      ;;
2531      ;;      get decimal number (0 <= N <= 255)
2532      ;;
2533      ;;      exit:      a = number
2534      ;;
2535      OC52 CD410A      gdn:      coll      skps          ;skip initial spaces
2536      OC55 2A6C0D      lhld      parsep        ;get pointer to current character
2537      OC58 229B0D      shld      errorp        ;save in case of parsing error
2538      OC5B C8          rz              ;return if end of command
2539      OC5C 7E          mov       a,m           ;get it
2540      OC5D FE30          cpi       '0'          ;error if non-numeric
2541      OC5F DA3A0C          jc       perror
2542      OC62 FE3A          cpi       '9'+1
2543      OC64 D23A0C          jnc      perror
2544      OC67 CD710C          call     gdns         ;convert number
2545      OC6A 226C0D      --      shld      parsep        ;save new position
2546      OC6D F601          ori       1           ;clear zero and carry flags
2547      OC6F 78          mov       a,b
2548      OC70 C9          ret
2549      ;
2550      OC71 0600      gdns:      mvi       b,0
2551      OC73 7E      gdns1:      mov       a,m
2552      OC74 D630          sui       '0'
2553      OC76 D8          rc
2554      OC77 FE0A          cpi       10
2555      OC79 D0          rnc
2556      OC7A F5          push      psw
2557      OC7B 78          mov       a,b           ;multiply current accumulator by 10
2558      OC7C 87          add       a
2559      OC7D 87          add       a
2560      OC7E 80          add       b
2561      OC7F 87          add       a
2562      OC80 47          mov       b,a
2563      OC81 F1          pop       psw
2564      OC82 23          inx       h           ;advance to next character
2565      OC83 80          add       b           ;add it in to the current accumulation
2566      OC84 47          mov       b,a
2567      OC85 FE10          cpi       16
2568      OC87 DA730C          jc       gdns1        ;loop unless >=16
2569      OC8A C33A0C          jmp       perror      ;error if invalid user number
2570      ;;
2571      ;;-----
2572      ;;
2573      ;;      print file name
2574      ;;
2575      if newdir
2576      OC8D 13      pfn:      inx       d           ;point to file name

```

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 570

PACIFIC GROVE, CA 93050

SER. # _____

```

2577 0C8E 2608      mvi    h,8          ;set # characters to print, clear # printed
2578 0C90 CD980C     call   pfn1        ;print name field
2579 0C93 CDA40C     call   space
2580 0C96 2603      mvi    h,3          ;set # characters to print
2581 0C98 1A        pfn1: ldax    d          ;get character
2582 0C99 E67F      ani     7fh
2583 0C9B CDA60C     call   pfc          ;print it if not
2584 0C9E 13        inx     d          ;advance pointer
2585 0C9F 25        dcr     h          ;loop if more to print
2586 0CA0 C2980C     jnz    pfn1
2587 0CA3 C9        ret
2588                ;
2589 0CA4 3E20      space: mvi    a,' '
2590                ;
2591 0CA6 C5        pfc:  push    b
2592 0CA7 D5        push    d
2593 0CA8 E5        push    h
2594 0CA9 CD1609     call   putc
2595 0CAC E1        pop     h
2596 0CAD D1        pop     d
2597 0CAE C1        pop     b
2598 0CAF C9        ret
2599
2600                else
2601
2602                pfn:  inx     d          ;point to file name
2603                lxi     b,8*256        ;set # characters to print, clear # printed
2604                call   pfn1        ;print name field
2605                ldax    d          ;see if there's a type
2606                ani     7fh
2607                cpi     ' '
2608                rz
2609                mvi     a,'.'        ;return if not
2610                call   pfc          ;print dot
2611                mvi     b,3          ;set # characters to print
2612                pfn1:  ldax    d          ;get character
2613                ani     7fh
2614                cpi     ' '          ;is it a space?
2615                cnz     pfc          ;print it if not
2616                inx     d          ;advance pointer
2617                dcr     b          ;loop if more to print
2618                jnz    pfn1
2619                ret
2620                ;
2621                space: mvi    a,' '
2622                ;
2623                pfc:  inr     c          ;increment # characters printed
2624                push    b
2625                push    d
2626                call   putc
2627                pop     d
2628                pop     b
2629                ret
2630                endif
2631                ;;
2632                ;;-----

```

CP/M - PLUS V3.0

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135

```

2633      ;;
2634      ;;      add a to hl
2635      ;;
2636      OCB0 85      addhla:      add      1
2637      OCB1 6F      mov      l,a
2638      OCB2 D0      rnc
2639      OCB3 24      inr      h
2640      OCB4 C9      ret
2641      ;;
2642      ;;-----
2643      ;;
2644      ;;      set match-any string into fcb
2645      ;;
2646      ;;      entry: de -> fcb area
2647      ;;      b = # bytes to set
2648      ;;
2649      ;;      setmatch:
2650      OCB5 3E3F      mvi      a,'?'      ;set match one character
2651      OCB7 12      setal: stax      d      ;fill rest of field with match one
2652      OCB8 13      inx      d
2653      OCB9 05      dcr      b      ;loop if more to fill
2654      OCBA C2B70C      jnz      setal
2655      OCB0 B7      ora      a
2656      OCBE C9      ret
2657      ;;
2658      ;;-----
2659      ;;
2660      ;;      table search
2661      ;;
2662      ;;      Search table of strings separated by spaces and terminated
2663      ;;      by 0. Accept abbreviations, but set string = matched string
2664      ;;      on exit so that we don't try to execute abbreviation.
2665      ;;
2666      ;;      entry: de -> string to search for
2667      ;;      hl -> table of strings to match (terminate table with 0)
2668      ;;      exit: z = set if match found
2669      ;;      a = entry # (0 thru n-1)
2670      ;;      z = not set if no match found
2671      ;;
2672      OCBF 01FF00      tbls: lxi      b,0ffh      ;clear entry & entry length counters
2673      OCC2 D5      tbls0: push      d      ;save match string addr
2674      OCC3 E5      push      h      ;save table string addr
2675      OCC4 1A      tbls1: ldax      d      ;compare bytes
2676      OCC5 E67F      ani      7fh      ;kill upper bit (so SYS + R/D match)
2677      OCC7 FE21      cpi      ' 't1      ;end of search string?
2678      OCC9 DAD00C      jc      tbls2      ;skip compare, if so
2679      OCC0 BE      cmp      m
2680      OCCD C2E00C      jnz      tbls3      ;jump if no match
2681      OCD0 13      tbls2: inx      d      ;advance string pointer
2682      OCD1 0C      inr      c      ;increment entry length counter
2683      OCD2 3E20      mvi      a,' '
2684      OCD4 BE      cmp      m
2685      OCD5 23      inx      h      ;advance table pointer
2686      OCD6 C2C40C      jnz      tbls1      ;continue with this entry if more
2687      OCD9 E1      pop      h      ;HL = matched string in table
2688      OCDA D1      pop      d      ;DE = string address

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #


```

2689  OCDB CDAE0B      call  move      ; C = length of string in table
2690  OCDE 78          mov    a,b      ;return current entry counter value
2691  OCDF C9          ret
2692                ;
2693  OCE0 3E20      tbls3: mvi    a,' '      ;advance hl past current string
2694  OCE2 BE      tbls4: cmp    m
2695  OCE3 23          inc    h
2696  OCE4 C2E20C     jnz    tbls4
2697  OCE7 D1          pop     d      ;throw away last table address
2698  OCE8 D1          pop     d      ;DE = string address
2699  OCE9 04          inc    b      ;increment entry counter
2700  OCEA 0EFF       mvi    c,0ffh
2701  OCEC 7E          mov    a,m      ;check for end of table
2702  OCED D601       sui    1
2703  OCEF D2C20C     jnc    tbls0      ;loop if more entries to test
2704  OCF2 C9          ret
2705                ;
2706                ;*****
2707                ;*****
2708                ;
2709                ;*****
2710                ;
2711                ;   DATA AREA
2712                ;
2713                ;*****
2714                ;   ;Note uninitialized data placed at the end (DS)
2715                ;
2716                ;
2717                if    prompts
2718                enter: db    'Enter $'
2719                unmsg: db    'User #: $'
2720                fnmsg: db    'File: $'
2721                else
2722  OCF3 456E746572unmsg: db    'Enter User #: $'
2723                endif
2724  OD02 4E6F204669nonmsg: db    'No Files'
2725                required:
2726  OD0A 2072657175    db    ' required$'
2727                ermsg:
2728  OD14 4552415345    db    'ERASE $'
2729                confirm:
2730  OD1B 2028592F4E    db    ' (Y/N)? $'
2731  OD24 0D0A0D0A50more: db    cr,lf,cr,lf,'Press RETURN to Continue $'
2732                if
2733  OD42 2020285573userzero db    ' (User 0)$'
2734                endif
2735                ;
2736                ;
2737                ;
2738                if    newdir
2739  OD4D 00          anyfiles: db    0      ;flag for SYS or DIR files exist
2740  OD4E 4E4F4E2D    dirfiles: db    'NON-'
2741  OD52 5359535445sysfiles: db    'SYSTEM FILE(S) EXIST$'
2742                endif
2743
2744  OD67 00          errflg:  db    0      ;parse error flag

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

2745             if multi
2746             multibuf1:
2747                 dw      0             ;multiple commands buffer length
2748             endif
2749             scbadd:   db      scbad-pag$off,0
2750             ;***** CAUTION FOLLOWING DATA MUST BE IN THIS ORDER *****
2751             pfncb:    ;BDOS func 152 (parse filename)
2752                 parsep: dw      0             ;pointer to current position in command
2753                 pfncb:  dw      pfcf          ;.fcb for func 152
2754                 usernum: ;CCP current user
2755                 db      0
2756             chaindisk:
2757                 db      0             ;transient's current disk
2758                 disk:   db      0             ;CCP current disk
2759                 subfcb: db      1,'$$$ SUB',0
2760             ccbpend:  ;end of file (on disk)
2761                 ds      1
2762                 submod: ds      1
2763                 subrc:  ds      1
2764                 ds      16
2765                 subcr:  ds      1
2766                 subrr:  ds      2
2767                 subrr2: ds      1
2768
2769             dircols:
2770                 ds      1             ;number of columns for DIR/DIRS
2771                 pgsize: ds      1             ;console page size
2772                 line:   ds      1             ;console line #
2773                 pgmode: ds      1             ;console page mode
2774             ;*****
2775                 errorp: ds      2             ;pointer to beginning of current param.
2776                 errsav: ds      2             ;pointer to built-in command tail
2777                 bufp:   ds      1             ;buffer pointer for getb
2778                 realdos:
2779                 ds      1             ;base page of BDOS
2780             ;
2781                 option: ds      1             ;'I' in line?
2782                 passwd: ds      10            ;password
2783                 ufcb:   ds      1             ;user number (must precede fcb)
2784             FCB:
2785                 ds      1             ; drive code
2786                 ds      8             ; file name
2787                 ds      3             ; file type
2788                 ds      4             ; control info
2789                 ds      16            ; disk map
2790                 fcbr:   ds      1             ; current record
2791                 fcbr:   ds      2             ; random record
2792                 pfcb:   ds      36            ; fcb for parsing
2793             ;
2794             ;
2795             ;
2796             ;
2797             ; command line buffer
2798             ;
2799                 cbufmx: ds      1
2800                 cbufl:  ds      1

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950
 SER. # _____

CP/M RMAC ASSEM 1.1 #051 CP/M 3 - CONSOLE COMMAND PROCESSOR - NOVEMBER 1982

2801	0DF6	cbuf:	ds	comlen
2802	0EDD		ds	50h
2803		stack:		
2804		ccptop:		;top page of CCP
2805	0F2D		end	

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

ADDHLA	0CB0	1200	1233	1422	1610	2373	2636#								
ALLOCF	0062	118#	313												
ANYFILES	0D4D	770	807	2739#											
AUXINRFLG	00C2	182#													
AUXOUTRFLG	00C4	183#													
BANKED	038F	24	36#	287											
BDOS	0005	72#	73	283	314	375	1335	1381	1391	1399	1418				
		1457	1469	1530	1600	1628	1915								
BDOSBASE	0098	158#	288												
BDOSF	0991	1001	1512#												
BDOSVERSION	00A1	161#	365												
BREAK	0966	826	883	911	1436#										
BREAK1	096D	1436	1440#	1825											
BUF	0080	83#	90	440	442	1264	1269	1455	1609	2372					
BUFP	0D9F	908	1625	2366	2371	2777#									
BUILTINERR	07BB	1012	1089#												
CALCDE	030F	22	34#	299											
CBUF	0DF6	1680	2801#												
CBUFL	0DF5	441	499	1114	1419	2800#									
CBUFMX	0DF4	1414	2799#												
CCP10	B4A0	239#	2401	2414											
CCPBDOS	B4B0	241#	564												
CCPBUILTIN	05B9	624#													
CCPCONBUF	00B1	170#	560	1803	1819	1822	1827								
CCPCR	0506	454	473#	842	890	904	1375	1438	2515						
CCPDISK0	05E8	630	641	646#	672										
CCPDISK1	05F9	652	655#												
CCPDISK2	0604	613	649	664#											
CCPDISK3	060A	620	670#												
CCPDRIVE	0625	676	684#												
CCPDRV	00AF	168#	692												
CCPEND	0D80	27	2760#												
CCPFLAG1	00B3	171#	381	1319											
CCPFLAG2	00B4	172#	370												
CCPFLAG3	00B5	173#	451												
CCPORG	041A	26	38#	44											
CCPPARSE	058B	447	529	583#											
CCPRET	050C	279	477#	482	1091										
CCPSUB	B420	240#													
CCPTOP	0F2D	2804#													
CCPUSER	061A	678#	932												
CCPUSR	00B0	169#	398	680											
CHAIN	04D9	441#	460												
CHAINDSK	0D71	1279	2756#												
CHAINENV	0040	226#	393												
CHAINFLG	0080	224#	438	1321											
CINF	0001	96#	1390	1443											
CKAFN	079E	947	1040#												
CKAFNO	07A0	1042#	1051												
CKBOOT	04E7	439	451#												
CLOSEF	0010	105#													
CLPDRV	00AB	165#	1109												
CLPFLGS	00AA	164#													
CNDRV	0050	74#	1253												
COLDBOOT	0002	260#	453	456											
COMBASE	00F9	217#	285												

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

COMLEN	00E7	86#	90#	1415	2801				
COMREDIRECT	B320	227#							
COMSIZE	0101	147#							
CONBUFFER	00BA	178#	560	1819					
CONBUFL	00BC	179#	1822						
CONCOLUMN	00B7	175#							
CONFIRM	0D1B	954	2729#						
CONINRFLG	00BE	180#							
CONLINE	00B9	177#							
CONMODE	00CF	191#	361	1309					
CONOUTRFLG	00C0	181#							
CONPAGE	00B8	176#	339						
CONTRAN	00CD	190#							
CONWIDTH	00B6	174#	329						
COPY0	0BB7	1267	1754	2344#					
COPY1	0BB9	2345#	2353						
COUTF	0002	97#	1380						
CR	000D	125#	1376	1781	1784	2180	2273	2461	2731
CRAWF	0006	98#							
CRLF	0C09	476	769	880	956	1188	1424	2461#	
CSTATF	000B	101#	503	1440					
CTBL	0637	625	707#						
CTLSTAT	CF01	264#							
CTLHACT	00CA	187#							
CTRLC	0003	124#	1374						
DATE	00F4	216#							
DATETIME	B404	250#							
DAYFILE	FFFF	55#	1170	2732					
DCNT	00E1	205#							
DEFDRV	0004	71#	1291						
DELF	0013	108#	519	535	962				
DFCB	005C	75#	76	738	783	998	1599	1627	1894
DFCB1	006C	79#	80	1274					
DIR	0675	715	753#	841#					
DIR0	06B3	792#							
DIR1	06B9	797#	830	854#	886				
DIR2	06CE	805	812#	861#					
DIR3	06E5	808	823#	858	879	882#			
DIRCOLS	0D97	337	793	2769#					
DIRDRV	0666	737#	817						
DIRDRV0	0669	741#	1175						
DIRDRV1	066D	554	745#						
DIRDRV2	0670	743	747#						
DIRECT	0699	764	779#						
DIREX	06F2	831#							
DIRFILES	0D4E	761	2740#						
DIRLN	0C0B	814	2460#						
DIRS	067D	719	759#	837#					
DIRS1	0682	756	763#						
DISK	0D72	691	746	1141	1534	2758#			
DISPLAY	B403	251#	1171						
DMAAD	00D8	198#							
DMAF	001A	112#	1456						
DRV0	00E8	210#	1140						
DRV1	00E9	211#							
DRV2	00EA	212#							

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

SER. #

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

GCMDS	0B9C	2021	2036#						
GCMDB	0B5B	1983	1990#	2017	2023	2025	2031		
GCMDB	0B5E	1995#	2041						
GDM	0C52	930	2535#						
GDNS	0C71	2012	2544	2550#					
GDNS1	0C73	2551#	2568						
GETB	0BC4	912	2365#						
GETBYTE	0BFF	330	1149	2440#					
GETC	0941	955	1373	1389#					
GETC1	0944	1390#							
GETFLG	0BDD	848	1172	2388#	2407	2419			
GETFN	0AD8	906	945	1888#					
GETPO	07AF	1064#							
GETPGMODE	0596	589	595#						
GETPRM	07AB	929	1061#	1892					
GFN	0AD8	782	844	972	981	1270	1894#		
GFNO	0ADB	1275	1895#						
GFN2	0AE4	1897	1909#	1998					
GFN3	0AFE	1924	1926#						
GFN4	0B09	1928	1931#						
GFN6	0B10	1933	1935#						
GFN7	0B31	1950#	1953						
GFN8	0B3B	1948	1954#						
GFNPWD	0B13	1936#							
HASH	009D	160#							
HASHL	009C	159#							
INFO	00DB	200#							
LENO	0053	78#	1273						
LEN1	0056	82#	1278						
LF	000A	126#	1360	1378	2463	2731	2731		
LINE	0B99	481	2772#						
LISTCP	00D4	195#							
LISTOUTRFLG	00C6	184#							
LOADER	090D	1322	1330#						
LOADF	003B	117#	1334						
LOCATION	0001	253#							
MAKEF	0016	110#							
MENU	B310	228#							
MODULE	0100	143#							
MORE	0B24	1372	2731#						
MOVE	0BAE	307	446	980	1113	1236	1256	1939	1942
		2689						2331#	2337
MOVLDR	043F	297#							
MULTCNT	00E6	208#	356						
MULTI	FFFF	58#	85	558	569	1710	1739	2315	2745
MULTIO	0A72	1775#	1783						
MULTI1	0A7D	1780	1782#						
MULTIBUFL	0D6B	2746#							
MULTIEND	0ACB	1828	1835	1841#					
MULTIRXPG	00AE	167#	1809	1847					
MULTISAVE	0AA8	571	1818#						
MULTISTART	0A59	1712	1741#						
NEWDIR	FFFF	53#	736	752	1077	2575	2738		
NEWERA	FFFF	54#	1045	1052					
NEXTADD	000B	136#	308						
NOFILE	07B8	766	1006	1087#					

COPYRIGHT © 1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

NOMSG	0D02	1088	2724#											
NOTCHAINFLG	003F	225#	1332											
OLOG	0090	156#	1311											
OPENCOM	098B	1162	1500#											
OPENF	000F	104#	501	909	1501									
OPTION	0DA1	631	1679	1702	2781#									
ORDER	B418	245#	647											
OSBASE	0006	73#	292	304	1770									
OUTDELIM	00D3	194#	350											
PAGEDEF	00C9	186#	591											
PAGEMODE	00C8	185#	596											
PAGOFF	009C	154#	156	157	158	159	160	161	162	163	164			
		165	166	167	168	169	170	171	172	173	174			
		175	176	177	178	179	180	181	182	183	184			
		185	186	187	188	189	190	191	192	194	195			
		196	197	198	199	200	201	202	203	204	205			
		206	207	208	209	210	211	212	213	214	215			
		216	217	218	219	2749								
PARSEF	0098	120#	1913											
PARSEP	0D6C	634	1008	1117	1262	1266	1698	1721	1722	1909	1935			
		1975	1994	2006	2536	2545	2752#							
PASS0	0051	77#	1271											
PASS1	0054	81#	1276											
PASSWD	0DA2	1521	1940	2782#										
PATCHFLAG	00ED	215#												
PBUFF	0009	99#	1398											
PDB	0C13	553	2470#											
PDB1	0C1A	2473#	2475											
PDB2	0C25	2471	2479#											
PERR2	0C4A	2513#												
PERROR	0C3A	666	1028	1930	2504#	2541	2543	2569						
PFC	0CA6	749	819	868	1177	2462	2464	2491	2583	2591#	2610			
		2615	2623#											
PFCB	0DD0	1937	2753	2792#										
PFN	0C8D	821	869	1010	1179	2576#	2602#							
PFN1	0C98	2578	2581#	2586	2604	2612#	2618							
PFNCB	0D6C	1912	2751#											
PFNFCB	0D6E	2753#												
PGNODE	0D9A	598	2773#											
PGSIZE	0D9B	1362	2771#											
PMSG	0949	950	1068	1071	1187	1389	1398#	2456						
PMSGNL	0C05	776	1090	2456#										
PREVADD	000C	137#												
PROFILE	04FC	459	461#											
PROGRETCODE	00AC	166#	1323	1689										
PROMPT	0564	488	550#											
PROMPTS	0000	56#	1065	1889	2717									
PSTR6	0C2A	953	2486#	2493	2512									
PTBL	065A	636	715#											
PUTC	0916	556	916	1360#	2480	2514	2594	2626						
PUTC0	0939	1367	1378#											
PUTC1	093B	1361	1379#											
PUTC2	093C	1377	1380#	2477										
QFLAG	00D5	196#												
RBUFF	000A	100#	1417											
RCLN	094E	567	1072	1414#										

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

READ	09E2	1624#	2368				
READF	0014	109#	1626				
REALDOS	0DA0	290	1248	1296	2778#		
RELOC	02CA	21	33#	303			
RELOG	00DE	202#					
REN	0751	718	972#				
RENF	0017	111#	985				
REQUIRED	0D0A	1011	2725#				
RESEL	00DD	201#					
RESEALLOC	0458	294	312#				
RESETCCPFLG	0BEE	568	1676	2413#			
RESETF	000D	102#	373				
RESETFLG	0BF1	534	565	2416#			
RLOG	0092	157#					
RREADF	0021	114#	515				
RSXCHAIN	0200	20	32#	385	1851		
RSXCK	049A	380#					
RSXEND	0394	25	37#	298			
RSXLEN	0112	149#					
RSXLOAD	0001	259#					
RSXOFF	0110	148#					
RSXONLYCLR	00FD	234#	388				
RSXONLYSET	0002	233#	383				
RSXSTART	0100	16	28#	298			
RUBOUTACT	00CB	188#					
RW	09EB	1441	1444	1628#	1641		
SBDOSF	076C	910	963	994#			
SBUF80	0978	781	1455#	1538			
SCB1	04BA	405	407#				
SCB2	04C4	414	417#				
SCBAD	00D6	197#	2749				
SCBADD	0D6A	281	2749#				
SCBADDR	038D	23	35#	284	1526	1802	2321
SCBF	0031	116#	282				
SCRINIT	045D	325#					
SEARCHA	00E3	206#					
SEARCHL	00E5	207#					
SEARF	0011	106#	1586				
SEARNF	0012	107#	1598				
SELDISK	00DA	199#	412	1289	1535		
SELECT	0980	689	1282	1466#	1520		
SELF	000E	103#	1468				
SETBUF	097B	500	1456#	1522			
SETBYTE	0BF9	681	693	1110	1310	1482	2434#
SETCCPFLG	0BE4	475	484	2400#			
SETFLG	0BE7	2404#					
SETM1	0CB7	2651#	2654				
SETMATCH	0CB5	787	849	2649#			
SETPGCODE	0591	590#					
SETTYPE	087A	657	660	1228#			
SETUSER	0986	682	1284	1480#			
SKPS	0A41	1062	1721#	1895	1967	2499	2535
SKPS1	0A44	1681	1722#	1732			
SKPS2	0A55	1728	1731#				
SPACE	0CA4	820	822	873	2579	2589#	2621#
SRCH	09CA	1587	1599#				

COPYRIGHT © 1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

SRCHF	09C3	789	1082	1586#					
SRCHN	09C8	827	884	1598#					
STACK	0F2D	278	478	2803#					
START	041A	276#							
SUBCLOSE	054F	505	524#						
SUBCR	0D93	2765#							
SUBFCB	0D73	1640	2759#						
SUBFILE	B301	231#	232	533					
SUBFILEMASK	0001	232#	487						
SUBKILL	0559	532#							
SUBMIT	B440	243#	2387						
SUBMITFLG	0040	244#							
SUBMOD	0D81	2762#							
SUBRC	0D82	512	2763#						
SUBRR	0D94	2766#							
SUBRR2	0D96	506	2767#						
SUBTEST	0BDA	588	2386#						
SUDOS	09F0	502	504	516	520	527	536	1640#	
SYSFILES	0D52	755	2741#						
TAB	0009	127#	1729	2264	2273				
TBLJ	0858	637	1199#						
TBL5	0CBF	629	1127	2672#					
TBL50	0CC2	2673#	2703						
TBL51	0CC4	2675#	2686						
TBL52	0CD0	2678	2681#						
TBL53	0CE0	2680	2693#						
TBL54	0CE2	2694#	2696						
TEMPDRV	00EC	214#	429						
TENBUFFER	00D1	192#							
TOPTPA	00FE	219#							
TPA	0100	84#	90	147	148	149	442	1246	
TRUE	FFFF	51#	53	54	55	57	58	2056	
TRUNF	0063	119#	526						
TRYCOM	0786	639	1009#						
TYPE	06F4	716	904#						
TYPE1	0705	911#	917						
TYPEAHEAD	00CC	189#							
TYPTBL	0863	1126	1209#	1232					
UC	09F6	604	1073	1675#					
UC0	0A1C	1686	1692	1696#					
UC1	0A1D	1688	1697#						
UC3	0A21	1699#	1716						
UC4	0A2A	1701	1703#						
UC5	0A37	1704	1706	1709#					
UFCB	0DAC	609	614	674	1129	2783#			
UNMSG	0CF3	928	2719#	2722#					
USER	0715	720	927#						
USERF	0020	113#							
USERNUM	0D70	399	551	679	1283	2704#			
USERPARSE	B304	230#							
USERZERO	0D42	1186	2733#						
USRCODE	00E0	204#	401	1481					
UTILFLGS	00A2	162#							
VERS	0031	95#	366						
WARNFLG	000E	138#	1796						
WORDMOV	0BA7	561	1820	1823	2320#				

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

WSTART	0000	70#
XCOM	0889	1214 1216 1242#
XCOM3	08A4	1262#
XCOM5	08B5	1270#
XCOM7	08CC	1279#
XCOM8	08E4	1295#
XCOM9	08EF	1308#
XFCB	07C1	1102# 1111
XPTBL	0870	1192 1214#
XSUB	07CD	1107# 1215
ZERO	0B26	1925 1929 1945#

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

1      title 'CP/M 3 - PROGRAM LOADER RSX - November 1982'
2      ;
3      ; version 3.0b Nov 04 1982 - Kathy Strutynski
4      ; version 3.0c Nov 23 1982 - Doug Huskey
5      ; Dec 22 1982 - Bruce Skidmore
6      ;
7      ; copyright (c) 1982
8      ; digital research
9      ; box 579
10     ; pacific grove, ca.
11     ; 93950
12     ;
13     *****
14     ***** The following values must be placed in ***
15     ***** equates at the front of CCP3.ASM. ***
16     *****
17     ***** Note: Due to placement at the front these ***
18     ***** equates cause PHASE errors which can be ***
19     ***** ignored. ***
20     P0100 = equ1 equ rsxstart +0100h ;set this equate in the CCP
21     P01D0 = equ2 equ fixchain +0100h ;set this equate in the CCP
22     P01EB = equ3 equ fixchain1+0100h ;set this equate in the CCP
23     P01F0 = equ4 equ fixchain2+0100h ;set this equate in the CCP
24     P0200 = equ5 equ rsx$chain+0100h ;set this equate in the CCP
25     P02CA = equ6 equ reloc +0100h ;set this equate in the CCP
26     P030F = equ7 equ calcdst +0100h ;set this equate in the CCP
27     P038D = equ8 equ scbaddr +0100h ;set this equate in the CCP
28     P038F = equ9 equ banked +0100h ;set this equate in the CCP
29     P0394 = equ10 equ rsxend +0100h ;set this equate in the CCP
30     P041A = ccporq equ CCP ;set origin to this in CCP
31     P0369 = patch equ patcharea+0100h ;LOADER patch area
32
33     041A = CCP equ 41Ah ;ORIGIN OF CCP3.ASM
34
35
36     *****
37
38     ; conditional assembly toggles:
39
40     FFFF = true equ 0ffffh
41     0000 = false equ 0h
42     FFFF = spacesaver equ true
43
44     0020 = stacksize equ 32 ;16 levels of stack
45     0030 = version equ 30h
46     0100 = tpa equ 100h
47     000F = ccptop equ 0Fh ;top page of CCP
48     0006 = osbase equ 06h ;base page in BDOS jump
49     000A = off$next equ 10 ;address in next jmp field
50     0020 = currec equ 32 ;current record field in fcb
51     0021 = ranrec equ 33 ;random record field in fcb
52
53
54
55
56

```

CP/M-PLUS V3.0

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135

```

57      ;      dsect for SCB
58      ;
59      0098 =      bdosbase      equ      98h          ; offset from page boundary
60      00B3 =      ccpflag1      equ      0b3h        ; offset from page boundary
61      00E6 =      multcnt       equ      0e6h        ; offset from page boundary
62      00FD =      rsxonly$clr   equ      0FDh        ;clear load RSX flag
63      0002 =      rsxonly$set   equ      002h
64      003A =      rscbadd       equ      3ah          ;offset of scbadd in SCB
65      003C =      dmaadd        equ      03ch        ;offset of DMA address in SCB
66      0062 =      bdosadd       equ      62h          ;offset of bdosadd in SCB
67      ;
68      0002 =      loadflag      equ      02H          ;flag for LOADER in memory
69      ;
70      ;      dsect for RSX
71      0006 =      entry         equ      06h          ;RSX contain jump to start
72      ;
73      000B =      nextadd       equ      0bh          ;address of next RXS in chain
74      000C =      prevadd       equ      0ch          ;address of previous RSX in chain
75      000E =      warmflg       equ      0eh          ;remove on wboot flag
76      0018 =      endchain      equ      18h          ;end of RSX chain flag
77      ;
78      ;
79      0014 =      readf         equ      20           ;sequential read
80      001A =      dmaf          equ      26           ;set DMA address
81      0031 =      scbf          equ      49           ;get/set SCB info
82      003B =      loadf         equ      59           ;load function
83      ;
84      ;
85      0040 =      maxread        equ      64           ;maximum of 64 pages in MULTIO
86      ;
87      ;
88      0000 =      wboot         equ      0000h        ;BIOS warm start
89      0005 =      bdos          equ      0005h        ;bdos entry point
90      0009 =      print         equ      9           ;bdos print function
91      000C =      vers          equ      12           ;get version number
92      0200 =      module        equ      200h         ;module address
93      ;
94      ;      DSECT for COM file header
95      ;
96      0101 =      comsize       equ      tpat1h
97      0103 =      scbcode       equ      tpat3h
98      0110 =      rsxoff        equ      tpat10h
99      0112 =      rsxlen        equ      tpat12h
100     ;
101     ;
102     000D =      cr             equ      0dh
103     000A =      lf             equ      0ah
104     ;
105     ;
106     ;      cseg
107     ;
108     ;
109     ;      ***** LOADER RSX HEADER *****
110     ;
111     rsxstart:
112     0000 C31A04      jmp      ccp          ;the ccp will move this loader to

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. # _____

```

113 0003 000000      db      0,0,0      ;high memory, these first 6 bytes
114                                     ;will receive the serial number from
115                                     ;the 6 bytes prior to the BDOS entry
116                                     ;point
117      tojump:
118 0006 C31800      jmp      begin
119 0009 C3          next db      0c3h      ;jump to next module
120 000A 0600      nextjmp dw      06
121 000C 0700      prevjmp dw      07
122 000E 00          db      0      ;warm start flag
123 000F 00          db      0      ;bank flag
124 0010 4C4F414445 db      'LOADER '    ;RSX name
125 0018 FF          db      0ffh      ;end of RSX chain flag
126 0019 00          db      0      ;reserved
127 001A 00          db      0      ;patch version number
128
129      ; ***** LOADER RSX ENTRY POINT *****
130
131      begin:
132 001B 79          mov      a,c
133 001C FE3B        cpi      loadf
134 001E C20900      jnz      next
135      beginlod:
136 0021 C1          pop      b
137 0022 C5          push     b      ;BC = return address
138 0023 210000      lxi      h,0      ;switch stacks
139 0026 39          dad      sp
140 0027 31BE02      lxi      sp,stack ;our stack
141 002A 229A02      shld     uestack ;save user stack address
142 002D C5          push     b      ;save return address
143 002E EB          xchg     ;save address of user's FCB
144 002F 229802      shld     usrfcb
145 0032 7C          mov      a,h      ;is .fcb = 0000h
146 0033 85          ora      l
147 0034 F5          push     psw
148 0035 CC0001      cz       rsx$chain ;if so , remove RSXs with remove flag on
149 0038 F1          pop      psw
150 0039 C43001      cnz     loadfile
151 003C D1          pop      d      ;return address
152 003D 210001      lxi      h,tpa
153 0040 7E          mov      a,m
154 0041 FEC9        cpi      ret
155 0043 CA9E00      jz       rsxfile
156 0046 7A          mov      a,d      ;check return address
157 0047 3D          dcr      a      ; if CCP is calling
158 0048 B3          ora      e      ; it will be 100H
159 0049 C25F00      jnz     retuser1 ;jump if not CCP
160      retuser:
161 004C 3A0D00      lda      prevjmp+1 ;get high byte
162 004F B7          ora      a      ;is it the zero page (i.e. no RSXs present)
163 0050 C25F00      jnz     retuser1 ;jump if not
164 0053 2A0A00      lhld     nextjmp ;restore five....don't stay around
165 0056 220600      shld     osbase
166 0059 229402      shld     newjmp
167 005C CDF800      call     setmaxb
168      retuser1:

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

169 005F 2A9A02      lhd      ustack      ;restore the stack
170 0062 F9          sphl
171 0063 AF          xra      a
172 0064 6F          mov      l,a
173 0065 67          mov      h,a      ;A,HL=0 (successful return)
174 0066 C9          ret          ;CCP pushed 100H on stack
175 ;
176 ;
177 ;      BDOS FUNC 59 error return
178 ;
179 reterror:
180 0067 11FE00      lxi      d,0feh
181 reterror1:
182 ;DE = BDOS error return
183 006A 2A9A02      lhd      ustack
184 006D F9          sphl
185 006E E1          pop      h      ;get return address
186 006F E5          push     h
187 0070 25          dcr      h      ;is it 100H?
188 0071 7C          mov      a,h
189 0072 B5          ora      l
190 0073 EB          xchg
191 0074 7D          mov      a,l      ;now HL = BDOS error return
192 0075 44          mov      b,h
193 0076 C0          rnz          ;return if not the CCP
194 ;
195 ;
196 loaderr:
197 0077 0E09          mvi      c,print
198 0079 115302      lxi      d,nogo      ;cannot load program
199 007C C00500      call     bdos      ;to print the message
200 007F C30000      jmp      wboot      ;warm boot
201 ;
202 ;
203 ;
204 ;
205 ;*****
206 ;
207 ;      MOVE RSXS TO HIGH MEMORY
208 ;
209 ;*****
210 ;
211 ;
212 ;      RSX files are present
213 ;
214 ;
215 0082 23          rsxf1: inx      h
216 0083 4E          mov      c,m
217 0084 23          inx      h
218 0085 46          mov      b,m      ;BC contains RSX length
219 0086 3A8F02      lda      banked
220 0089 B7          ora      a      ;is this the non-banked system?
221 008A CA9200      jz      rsxf2      ;jump if so
222 008D 23          inx      h      ;HL = banked/non-banked flag
223 008E 34          inr      m      ;is this RSX only for non-banked?
224 008F CA9D00      jz      rsxf3      ;skip if so

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____


```

225 0092 D5      rsxf2: push    d          ;save offset
226 0093 CD0F02  call    calcdst      ;calculate destination address and bias
227 0096 E1      pop     h          ;rsx offset in file
228 0097 CDA01   call    reloc       ;move and relocate file
229 009A CDD000   call    fixchain     ;fix up rsx address chain
230 009D E1      rsxf3: pop     h          ;RSX length field in header
231
232
233
234              rsxfile:
235              ;HL = .RSX (n-1) descriptor
236 009E 111000   lxi     d,10h          ;length of RSX descriptor in header
237 00A1 19      dad     d          ;HL = .RSX (n) descriptor
238 00A2 E5      push    h          ;RSX offset field in COM header
239 00A3 5E      mov     e,m
240 00A4 23      inx     h          ;DE = RSX offset
241 00A5 56      mov     d,m
242 00A6 7B      mov     a,e
243 00A8 C2B200   jnz     rsxf1      ;jump if RSX offset is non-zero
244 ;
245 ;
246 ;
247 comfile:
248              ;RSXs are in place, now call SCB setting code
249 00AB CD0301   call    scbcode      ;set SCB flags for this com file
250              ;is there a real COM file?
251 00AE 3A0002   lda     module      ;is this an RSX only
252 00B1 FEC9     cpi     ret
253 00B3 C2BF00   jnz     comfile2     ;jump if real COM file
254 00B6 2A8D02   lhld    scbaddr
255 00B9 2EB3     mvi     l,ccpflag1
256 00BB 7E      mov     a,m
257 00BC F602     ori     rsx$only$set ;set if RSX only
258 00BE 77      mov     m,a
259 comfile2:
260 00BF 2A0101   lhld    comsize      ;move COM module to 100H
261 00C2 44      mov     b,h
262 00C3 4D      mov     c,l          ;BC contains length of COM module
263 00C4 210002   lxi     h,tpa+100h      ;address of source for COM move to 100H
264 00C7 110001   lxi     d,tpa          ;destination address
265 00CA CD2602   call    move
266 00CD C35F00   jmp     retuser1      ;restore stack and return
267 ;;
268 ;*****
269 ;
270 ; ADD AN RSX TO THE CHAIN
271 ;
272 ;*****
273 ;
274 ;
275 fixchain:
276 00D0 2A0600   lhld    osbase      ;next RSX link
277 00D3 2E00     mvi     l,0
278 00D5 010600   lxi     b,6
279 00D8 CD2602   call    move          ;move serial number down
280 00DB 1E18     mvi     e,endchain

```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```

281 00DD 12          stax      ;set loader flag=0
282 00DE 1E0D        mvi      e,prevadd+1
283 00E0 12          stax      d          ;set previous field to 0007H
284 00E1 1B          dcx       d
285 00E2 3E07        mvi      a,7
286 00E4 12          stax      d          ;low byte = 7H
287 00E5 6B          mov       l,e        ;HL address previous field in next RSX
288 00E6 1E0B        mvi      e,nextadd   ;change previous field in link
289 00EB 73          mov       m,e
290 00E9 23          inx       h
291 00EA 72          mov       m,d        ;current <-- next
292
293 fixchain1:
294          ;entry: H=next RSX page,
295          ;      DE=(high byte of next RSX field) in current RSX
296 00EB EB          xchg      ;HL-->current DE-->next
297 00EC 72          mov       m,d        ;put page of next RSX in high(next field)
298 00ED 2B          dcx       h
299 00EE 3606        mvi      m,6
300
301 fixchain2:
302          ;entry: H=page of lowest active RSX in the TPA
303          ;this routine resets the BDOS address @ 6H and in the SCB
304 00F0 2E06        mvi      l,6
305 00F2 220600       shld      osbase      ;change base page BDOS vector
306 00F5 229402       shld      newjmp      ;change SCB value for BDOS vector
307
308
309 setmaxb:
310 00F8 119202       lxi      d,scbadd2
311 scbfun:
312 00FB 0E31        mvi      c,scbf
313 00FD C30500       jmp      bdos
314
315
316
317
318
319          ;      REMOVE TEMPORARY RSXS
320
321
322
323
324
325 rsx$chain:
326          ;
327          ;      Chase up RSX chain, removing RSXS with the
328          ;      remove flag on (OFFH)
329          ;
330 0100 2A0600       lhl      osbase      ;base of RSX chain
331 0103 44          mov       b,h
332
333 rsx$chain1:
334          ;B = current RSX
335 0104 60          mov       h,b
336 0105 2E18        mvi      l,endchain

```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```

337 0107 34      inr     m
338 0108 35      dcr     m                ;is this the loader?
339 0109 C0      rnz     m                ;return if so (m=0ffh)
340 010A 2E0B     mvi     l,nextadd       ;address of next node
341 010C 46      mov     b,m             ;DE -> next link
342              ;
343              ;
344      check$remove:
345              ;
346 010D 2E0E     mvi     l,warmflag       ;check remove flag
347 010F 7E      mov     a,m             ;warmflag in A
348 0110 B7      ora     a               ;FF if remove on warm start
349 0111 CA0401   jz      rsx$chain1       ;check next RSX if not
350              ;
351      remove:
352              ;remove this RSX from chain
353              ;
354              ;first change next field of prior link to point to next RSX
355              ;HL = current B = next
356              ;
357 0114 2E0C     mvi     l,prevadd        ;address of previous RSX link
358 0116 5E      mov     e,m
359 0117 23      inx     h
360 0118 56      mov     d,m
361 0119 78      mov     a,b             ;A = next (high byte)
362 011A 12      stax    d               ;store in previous link
363 011B 1B      dcx     d               ;previous RSX chains to next RSX
364 011C 3E06     mvi     a,6             ;initialize low byte to 6
365 011E 12      stax    d               ;
366 011F 13      inx     d               ;DE = .next (high byte)
367              ;
368              ;now change previous field of next link to address previous RSX
369 0120 60      mov     h,b             ;next in HL...previous in DE
370 0121 2E0C     mvi     l,prevadd
371 0123 73      mov     a,e
372 0124 23      inx     h
373 0125 72      mov     m,d             ;next chained back to previous RSX
374 0126 7A      mov     a,d             ;check to see if this is the bottom
375 0127 B7      ora     a               ;RSX...
376 0128 C5      push    b
377 0129 CCF000   cz      fixchain2       ;reset BDOS BASE to page in H
378 012C C1      pop     b
379 012D C30401   jmp     rsx$chain1       ;check next RSX in the chain
380              ;
381              ;
382              ;
383      ;*****
384      ;
385      ;      PROGRAM LOADER
386      ;
387      ;*****
388      ;
389      ;
390      ;
391      loadfile:
392      ;      entry: HL = ,FCB

```

COPYRIGHT ©1982

DIGITAL RESEARCH

P.O. BOX 579

PACIFIC GROVE, CA 93950

SER. #

```

393 0130 E5      push    h
394 0131 119002   lxi     d,scbdma
395 0134 CDFB00   call    scbfun
396 0137 EB      xchg
397 0138 E1      pop     h           ;.fcb
398 0139 E5      push    h           ;save .fcb
399 013A 012000   lxi     b,currec
400 013D 09      dad     b
401 013E 3600     nvi     m,0           ;set current record to 0
402 0140 23      inx     h
403 0141 4E      mov     c,m           ;load address
404 0142 23      inx     h
405 0143 66      mov     h,m
406 0144 69      mov     l,c
407 0145 25      dcr     h
408 0146 24      inr     h
409 0147 CA6700   jz      reterror           ;Load address < 100h
410 014A E5      push    h           ;now save load address
411 014B D5      push    d           ;save the user's DMA
412 014C E5      push    h
413 014D CD3102   call    multiol           ;returns A=multio
414 0150 E1      pop     h
415 0151 F5      push    psw           ;save A = user's multisector I/O
416 0152 1E80     mvi     e,l28           ;read 16K
417
418             ;stack:      ;return address;
419             ;           ;.FCB           ;
420             ;           ;Load address   ;
421             ;           ;users DMA      ;
422             ;           ;users Multio   ;
423             ;
424
425 loadf0:
426             ;HL= next load address (DMA)
427             ; E= number of records to read
428 0154 3A0700     lda     osbase+1           ;calculate maximum number of pages
429 0157 3D      dcr     a
430 0158 94      sub     h
431 0159 DAFA01     jc      endload           ;we have used all we can
432 015C 3C      inr     a
433 015D FE40     cpi     maxread           ;can we read 16K?
434 015F D27601     jnc     loadf2
435 0162 07      rlc
436 0163 5F      mov     e,a           ;change to sectors
437 0164 7D      mov     a,l           ;save for multi i/o call
438 0165 B7      ora     a           ;A = low(load address)
439 0166 CA7601     jz      loadf2           ;load on a page boundary
440 0169 0602     mvi     b,2           ;((to subtract from # of sectors)
441 016B 3D      dcr     a           ;is it greater than 81h?
442 016C FA7001     jm      subtract           ;080h < l(adr) <= 0FFh (subtract 2)
443 016F 05      dcr     b           ;000h < l(adr) <= 080h (subtract 1)
444 subtract:
445 0170 7B      mov     a,e           ;reduce the number of sectors to
446 0171 90      sub     b           ;compensate for non-page aligned
447             ;load address
448 0172 CAFA01     jz      endload           ;can't read zero sectors

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

449 0175 5F      mov     e,a
450              ;
451              loadf2:
452              ;read the file
453 0176 D5      push    d                ;save number of records to read
454 0177 E5      push    h                ;save load address
455 0178 CD3302   call    multio          ;set multi-sector i/o
456 017B E1      pop     h
457 017C E5      push    h
458 017D CD3B02   call    readb          ;read sector
459 0180 E1      pop     h
460 0181 D1      pop     d                ;restore number of records
461 0182 F5      push    psw             ;zero flag set if no error
462 0183 7B      mov     a,e             ;number of records in A
463 0184 3C      inr     a
464 0185 1F      rar     a                ;convert to pages
465 0186 84      add     h
466 0187 67      mov     h,a             ;add to load address
467 0188 229602   shld    loadtop        ;save next free page address
468 018B F1      pop     psw
469 018C CA5401   jz      loadf0          ;loop if more to go
470
471              loadf4:
472              ;FINISHED load A=1 if successful (eof)
473              ; A>1 if a I/O error occured
474              ;
475 018F C1      pop     b                ;B=multisector I/O count
476 0190 3D      dcr     a                ;not eof error?
477 0191 58      mov     e,b             ;user's multisector count
478 0192 CD3302   call    multio
479 0195 0E1A     mvi     c,dmaf         ;restore the user's DMA address
480 0197 D1      pop     d
481 0198 F5      push    psw             ;zero flag => successful load
482 0199 CD0500   call    bdos           ; user's DMA now restored
483 019C F1      pop     psw
484 019D 2A9C02   lhld    bdosret        ;BDOS error return
485 01A0 EB      xchg
486 01A1 C26A00   jnz     reterror1
487 01A4 D1      pop     d                ;load address
488 01A5 E1      pop     h                ;.fcb
489 01A6 010900   lxi     b,9            ;is it a PRL?
490 01A9 09      dad     b                ;.fcb(type)
491 01AA 7E      mov     a,m
492 01AB E67F     ani     7fh            ;get rid of attribute bit
493 01AD FE50     cpi     'P'            ;is it a P?
494 01AF C0      rnz
495 01B0 23      inx     h                ;return if not
496 01B1 7E      mov     a,m
497 01B2 E67F     ani     7fh
498 01B4 FE52     cpi     'R'            ;is it a R
499 01B6 C0      rnz                    ;return if not
500 01B7 23      inx     h
501 01B8 7E      mov     a,m
502 01B9 E67F     ani     7fh
503 01BB D64C     sui     'L'            ;is it a L?
504 01BD C0      rnz                    ;return if not

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

505                ;load PRL file
506 01BE 7B        mov    a,e
507 01BF B7        ora     a                ;is load address on a page boundary
508 01C0 C26700    jnz     reterror        ;error, if not
509 01C3 62        mov     h,d
510 01C4 6B        mov     l,e                ;HL,DE = load address
511 01C5 23        inc     h
512 01C6 4E        mov     c,m
513 01C7 23        inc     h
514 01C8 46        mov     b,m
515 01C9 6B        mov     l,e                ;HL,DE = load address BC = length
516                ;         jmp     reloc        ;relocate PRL file at load address
517                ;
518                ;;
519                ;*****
520                ;
521                ;     PAGE RELOCATOR
522                ;
523                ;*****
524                ;
525                ;
526 reloc:
527                ;     HL,DE = load address (of PRL header)
528                ;     BC  = length of program (offset of bit map)
529 01CA 24        inc     h                ;offset by 100h to skip header
530 01CB 05        push    d                ;save destination address
531 01CC C5        push    b                ;save length in bc
532 01CD CD2602    call    move            ;move rsx to correct memory location
533 01D0 C1        pop     b
534 01D1 D1        pop     d
535 01D2 05        push    d                ;save DE for fixchain...base of RSX
536 01D3 5A        mov     e,d            ;E will contain the BIAS from 100h
537 01D4 1D        dec     e                ;base address is now 100h
538                ;                     ;after move HL addresses bit map
539                ;
540                ;storage moved, ready for relocation
541                ;     HL addresses beginning of the bit map for relocation
542                ;     E contains relocation bias
543                ;     D contain relocation address
544                ;     BC contains length of code
545 01D5 E5        rel0: push    h                ;save bit map base in stack
546 01D6 63        mov     h,e                ;relocation bias is in e
547 01D7 1E00      mvi     e,0
548                ;
549 01D9 78        rel1: mov     a,b            ;bc=0?
550 01DA B1        ora     c
551 01DB CAF701    jz      endrel
552                ;
553                ;     not end of the relocation, may be into next byte of bit map
554 01DE 0B        dcx     b                ;count length down
555 01DF 7B        mov     a,e
556 01E0 E607      ani     111b            ;0 causes fetch of next byte
557 01E2 C2EA01    jnz     rel2
558                ;     fetch bit map from stacked address
559 01E5 E3        xthl
560 01E6 7E        mov     a,m            ;next 8 bits of map

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P.O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

561 01E7 23      inx      h
562 01EB E3      xthl             ;base address goes back to stack
563 01E9 6F      mov      l,a      ;l holds the map as we process 8 locations
564 01EA 7D      rel2:  mov      a,l
565 01EB 17      ral             ;cy set to 1 if relocation necessary
566 01EC 6F      mov      l,a      ;back to l for next time around
567 01ED D2F301   jnc      rel3     ;skip relocation if cy=0
568              ;
569              ;      current address requires relocation
570 01F0 1A      ldax      d
571 01F1 84      add      h      ;apply bias in h
572 01F2 12      stax      d
573 01F3 13      rel3:  inx      d      ;to next address
574 01F4 C3D901   jmp      rel1     ;for another byte to relocate
575              ;
576              ;      end of relocation
577 01F7 D1      pop      d      ;clear stacked address
578 01F8 D1      pop      d      ;restore DE to base of PRL
579 01F9 C9      ret

```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

```

580
581
582              ;
583              ;;
584              ;*****
585              ;
586              ;      PROGRAM LOAD TERMINATION
587              ;
588              ;*****
589              ;
590              ;;
591              ;;
592      endload:
593 01FA CD3102     call      multio1      ;try to read after memory is filled
594 01FD 218000     lxi      h,80h        ;set load address = default buffer
595 0200 CD3B02     call      readb
596 0203 C28F01     jnz      loadf4       ;eof => successful
597 0206 21FE00     lxi      h,0feh       ;set BDOSRET to indicate an error
598 0209 229C02     shld     bdosret
599 020C C38F01     jmp      loadf4       ;unsuccessful (file to big)
600              ;
601              ;;
602              ;
603              ;;
604              ;*****
605              ;
606              ;      SUBROUTINES
607              ;
608              ;*****
609              ;
610              ;
611              ;
612              ;      Calculate RSX base in the top of the TPA
613              ;
614      calcdst:
615              ;
616              ;      calcdst returns destination in DE

```

```

617      ;      BC contains length of RSX
618      ;
619 020F 3A0700      lda      osbase+1      ;a has high order address of memory top
620 0212 3D          dcr      a              ;page directly below bdos
621 0213 0B          dcx      b              ;subtract 1 to reflect last byte of code
622 0214 90          sub      b              ;a has high order address of reloc area
623 0215 03          inx      b              ;add 1 back get bit map offset
624 0216 FE0F        cpi      ccptop        ;are we below the CCP
625 0218 DA7700      jc       loaderr
626 021B 2A9602      lhld     loadtop
627 021E BC          cmp      h              ;are we below top of this module
628 021F DA7700      jc       loaderr
629 0222 57          mov      d,a
630 0223 1E00        mvi      e,0            ;d,e addresses base of reloc area
631 0225 C9          ret
632      ;
633      ;
634      ;-----
635      ;
636      ;      move memory routine
637
638 move:
639      ;      move source to destination
640      ;      where source is in HL and destination is in DE
641      ;      and length is in BC
642      ;
643 0226 78          mov      a,b      ;bc=0?
644 0227 B1          ora      c
645 0228 C8          rz
646 0229 0B          dcx      b      ;count module size down to zero
647 022A 7E          mov      a,m      ;get next absolute location
648 022B 12          stax     d      ;place it into the reloc area
649 022C 13          inx      d
650 022D 23          inx      h
651 022E C32602      jmp      move
652      ;
653      ;-----
654      ;
655      ;      Multi-sector I/O
656      ;      (BDOS function #44)
657      ;
658 multiol:
659 0231 1E01        mvi      e,1            ;set to read 1 sector
660      ;
661 multio:
662      ;entry: E = new multisector count
663      ;exit:  A = old multisector count
664 0233 2A8D02      lhld     sbaddr
665 0236 2EE6        mvi      l,multicnt
666 0238 7E          mov      a,m
667 0239 73          mov      m,e
668 023A C9          ret
669      ;
670      ;-----
671      ;
672      ;      read file

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____


```

673      ;;      (BDOS function #20)
674      ;;
675      ;;      entry: hl = buffer address (readb only)
676      ;;      exit   z = set if read ok
677      ;;
678 023B EB      readb: xchg
679 023C 0E1A      setbuf: mvi    c,dmaof
680 023E E5                push    h                ;save number of records
681 023F CD0500          call    bdos
682 0242 0E14            mvi    c,readf
683 0244 2A9802          lhl    usrfcb
684 0247 EB            xchg
685 0248 CD0500          call    bdos
686 024B 229C02          shld    bdosret            ;save bdos return
687 024E D1            pop     d                ;restore number of records
688 024F B7            ora     a
689 0250 C8            rz                        ;no error on read
690 0251 5C            mov     e,h                ;change E to number records read
691 0252 C9            ret
692      ;
693      ;
694      ;*****
695      ;
696      ;      DATA AREA
697      ;
698      ;*****
699      ;
700
701 0253 0D0A43616Enogo  db      cr,lf,'Cannot load Program$'
702
703      patcharea:
704 0269                ds      36                ;36 byte patch area
705
706 028D 0000          scbaddr  dw      0
707 028F 00          banked  db      0
708
709 0290 3C          scbdma  db      dmaod
710 0291 00                db      00h                ;getting the value
711 0292 62          scbadd2  db      bdosadd            ;current top of TPA
712 0293 FE                db      0feh                ;set the value
713      ;
714
715      if not spacesaver
716
717      newjmp ds      2                ;new BDOS vector
718      loadtop        ds      2                ;page above loaded program
719      usrfcb ds      2                ;contains user FCB add
720      ystack:        ds      2                ; user stack on entry
721      bdosret        ds      2                ;bdos error return
722      ;
723      rsxend :
724      stack equ      rsxend+stacksize
725
726      else
727
728      rsxend:

```

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```
729 0294 = newjmp equ rsxend
730 0296 = loadtop equ rsxend+2
731 0298 = usrfcb equ rsxend+4
732 029A = ystack equ rsxend+6
733 029C = bdosret equ rsxend+8
734 02BE = stack equ rsxend+10+stacksize
735
736 endif
737 0294 end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. # _____

BANKED	028F	28	219	707#		
BDOS	0005	89#	199	313	482	681 685
BDOSADD	0062	66#	711			
BDOSBASE	0098	59#				
BDOSRET	029C	484	598	686	721#	733#
BEGIN	001B	118	131#			
BEGINLOD	0021	135#				
CALCDEST	020F	26	226	614#		
CCP	041A	30	33#	112		
CCPFLAG1	00B3	60#	255			
CCPORG	0000	30#				
CCPTOP	000F	47#	624			
CHECKREMOVE	010D	344#				
COMFILE	00AB	247#				
COMFILE2	00BF	253	259#			
COMSIZE	0101	96#	260			
CR	000D	102#	701			
CURREC	0020	50#	399			
DMAAD	003C	65#	709			
DMAF	001A	80#	479	679		
ENDCHAIN	0018	76#	280	336		
ENDLOAD	01FA	431	448	592#		
ENDREL	01F7	551	576#			
ENTRY	0006	71#				
EQU1	0000	20#				
EQU10	0000	29#				
EQU2	0000	21#				
EQU3	0000	22#				
EQU4	0000	23#				
EQU5	0000	24#				
EQU6	0000	25#				
EQU7	0000	26#				
EQU8	0000	27#				
EQU9	0000	28#				
FALSE	0000	41#				
FIXCHAIN	00D0	21	229	275#		
FIXCHAIN1	00EB	22	293#			
FIXCHAIN2	00F0	23	301#	377		
LF	000A	103#	701			
LOADERR	0077	196#	625	628		
LOADF	0038	82#	133			
LOADF0	0154	425#	469			
LOADF2	0176	434	439	451#		
LOADF4	018F	471#	596	599		
LOADFILE	0130	150	391#			
LOADFLAG	0002	68#				
LOADTOP	0296	467	626	718#	730#	
MAXREAD	0040	85#	433			
MODULE	0200	92#	251			
MOVE	0226	265	279	532	638#	651
MULTICNT	00E6	61#	665			
MULTIO	0233	455	478	661#		
MULTIO1	0231	413	593	658#		
NEWJMP	0294	166	306	717#	729#	
NEXT	0009	119#	134			
NEXTADD	000B	73#	288	340		

COPYRIGHT ©1982
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950
 SER. # _____

NEXTJMP	000A	120#	164						
NOGO	0253	198	701#						
OFFNXT	000A	49#							
OSBASE	0006	48#	165	276	305	330	428	619	
PATCH	0000	31#							
PATCHAREA	0269	31	703#						
PREVADD	000C	74#	282	357	370				
PREVJMP	000C	121#	161						
PRINT	0009	90#	197						
RANREC	0021	51#							
READB	023B	458	595	678#					
READF	0014	79#	682						
REL0	01D5	545#							
REL1	01D9	549#	574						
REL2	01EA	557	564#						
REL3	01F3	567	573#						
RELOC	01CA	25	228	526#					
REMOVE	0114	351#							
RETERROR	0067	179#	409	508					
RETERROR1	006A	181#	486						
RETUSER	004C	160#							
RETUSER1	005F	159	163	168#	266				
RSCBADD	003A	64#							
RSXCHAIN	0100	24	148	325#					
RSXCHAIN1	0104	333#	349	379					
RSXEND	0294	29	723#	724	728#	729	730	731	732
RSXF1	0082	215#	243						
RSXF2	0092	221	225#						
RSXF3	009D	224	230#						
RSXFILE	009E	155	233#						
RSXLEN	0112	99#							
RSXOFF	0110	98#							
RSXONLYCLR	00FD	62#							
RSXONLYSET	0002	63#	257						
RSXSTART	0000	20	111#						
SCBADD2	0292	310	711#						
SCBADDR	028D	27	254	664	706#				
SCBCODE	0103	97#	249						
SCBDMA	0290	394	709#						
SCBF	0031	81#	312						
SCBFUN	00FB	311#	395						
SETBUF	023C	679#							
SETHAXB	00F8	167	309#						
SPACESAVER	FFFF	42#	715						
STACK	02BE	140	724#	734#					
STACKSIZE	0020	44#	724	734					
SUBTRACT	0170	442	444#						
TOJUMP	0006	117#							
TPA	0100	46#	96	97	98	99	152	263	264
TRUE	FFFF	40#	42						
USRFCB	0298	144	683	719#	731#				
USTACK	029A	141	169	183	720#	732#			
VERS	000C	91#							
VERSION	0030	45#							
WARNFLG	000E	75#	346						
WBOOT	0000	88#	200						

CP/M - PLUS V3.0

COPYRIGHT ©1982

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA 93950

SER. # CP3-135